REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
26-03-2007	Final Technical Report	From 01-06-2003 to 31-12-2006

4. TITLE AND SUBTITLE

Intelligent Control for Future Autonomous Distributed Sensor Systems

5a. CONTRACT NUMBER

5b. GRANT NUMBER

N00014-03-1-0751

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Jannett, Thomas, C.

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

The University of Alabama at Birmingham
Office of Grants and Contracts
Birmingham, Alabama 35294

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Office of Naval Research
Ballston Centre Tower One
800 North Quincy Street
Arlington, VA 22217-5660

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution is Unlimited

13. SUPPLEMENTARY NOTES

14. ABSTRACT

Key distributed coordination and control technologies needed for future deployable autonomous distributed systems were investigated. Specific tasks accomplished include: 1) identification and exploration of alternative distributed intelligent system architectures, methods for distributed data fusion, control, and coordination, and strategies that will increase the ability of the field to survive attacks, failures, and accidents, 2) development of new intelligent software agents that facilitate distributed coordination and control, and 3) creation of computer simulations for developing and testing alternative algorithms, considering tradeoffs, and evaluating the ability of the distributed system to achieve its goals when presented with a set of operational challenges. This approach allowed design alternatives to be explored, allowed trade-offs to be exposed, provided insight into the parameters that impact overall system performance, and facilitated the identification of requirements for further development.

15. SUBJECT TERMS

agents, detection, localization, intelligent systems, reconfiguration, resource management, sensor networks, simulation

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE	SAR	100	Principal Investigator: Jannett, Thomas, C.
					19b. TELEPHONE NUMBER *(Include area code)* 205-934-8440

INSTRUCTIONS FOR COMPLETING SF 298

1. REPORT DATE. Full publication date, including day, month, if available. Must cite at least the year and be Year 2000 compliant, e.g. 30-06-1998; xx-06-1998; xx-xx-1998.

2. REPORT TYPE. State the type of report, such as final, technical, interim, memorandum, master's thesis, progress, quarterly, research, special, group study, etc.

3. DATES COVERED. Indicate the time during which the work was performed and the report was written, e.g., Jun 1997 - Jun 1998; 1-10 Jun 1996; May - Nov 1998; Nov 1998.

4. TITLE. Enter title and subtitle with volume number and part number, if applicable. On classified documents, enter the title classification in parentheses.

5a. CONTRACT NUMBER. Enter all contract numbers as they appear in the report, e.g. F33615-86-C-5169.

5b. GRANT NUMBER. Enter all grant numbers as they appear in the report, e.g. AFOSR-82-1234.

5c. PROGRAM ELEMENT NUMBER. Enter all program element numbers as they appear in the report, e.g. 61101A.

5d. PROJECT NUMBER. Enter all project numbers as they appear in the report, e.g. 1F665702D1257; ILIR.

5e. TASK NUMBER. Enter all task numbers as they appear in the report, e.g. 05; RF0330201; T4112.

5f. WORK UNIT NUMBER. Enter all work unit numbers as they appear in the report, e.g. 001; AFAPL30480105.

6. AUTHOR(S). Enter name(s) of person(s) responsible for writing the report, performing the research, or credited with the content of the report. The form of entry is the last name, first name, middle initial, and additional qualifiers separated by commas, e.g. Smith, Richard, J, Jr.

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES). Self-explanatory.

8. PERFORMING ORGANIZATION REPORT NUMBER. Enter all unique alphanumeric report numbers assigned by the performing organization, e.g. BRL-1234; AFWL-TR-85-4017-Vol-21-PT-2.

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES). Enter the name and address of the organization(s) financially responsible for and monitoring the work.

10. SPONSOR/MONITOR'S ACRONYM(S). Enter, if available, e.g. BRL, ARDEC, NADC.

11. SPONSOR/MONITOR'S REPORT NUMBER(S). Enter report number as assigned by the sponsoring/ monitoring agency, if available, e.g. BRL-TR-829; -215.

12. DISTRIBUTION/AVAILABILITY STATEMENT. Use agency-mandated availability statements to indicate the public availability or distribution limitations of the report. If additional limitations/ restrictions or special markings are indicated, follow agency authorization procedures, e.g. RD/FRD, PROPIN, ITAR, etc. Include copyright information.

13. SUPPLEMENTARY NOTES. Enter information not included elsewhere such as: prepared in cooperation with; translation of; report supersedes; old edition number, etc.

14. ABSTRACT. A brief (approximately 200 words) factual summary of the most significant information.

15. SUBJECT TERMS. Key words or phrases identifying major concepts in the report.

16. SECURITY CLASSIFICATION. Enter security classification in accordance with security classification regulations, e.g. U, C, S, etc. If this form contains classified information, stamp classification level on the top and bottom of this page.

17. LIMITATION OF ABSTRACT. This block must be completed to assign a distribution limitation to the abstract. Enter UU (Unclassified Unlimited) or SAR (Same as Report). An entry in this block is necessary if the abstract is to be limited.

INTELLIGENT CONTROL FOR FUTURE DEPLOYABLE AUTONOMOUS DISTRIBUTED SENSOR SYSTEMS

FINAL TECHNICAL REPORT

AWARD NUMBER: N00014-03-1-0751
PERIOD OF PERFORMANCE: 01-Jun-2003 to 31-Dec-2006

Principal Investigator: Thomas C. Jannett, Ph.D.

Thomas C. Jannett, Ph.D.
Professor of Electrical and Computer Engineering
The University of Alabama at Birmingham
BEC 259D
1530 3rd Ave S
Birmingham, Alabama 35294
205-934-8440
tjannett@uab.edu

ABSTRACT

Key distributed coordination and control technologies needed for future deployable autonomous distributed systems were investigated. Specific tasks accomplished include: 1) identification and exploration of alternative distributed intelligent system architectures, methods for distributed data fusion, control, and coordination, and strategies that will increase the ability of the field to survive attacks, failures, and accidents, 2) development of new intelligent software agents that facilitate distributed coordination and control, and 3) creation of computer simulations for developing and testing alternative algorithms, considering tradeoffs, and evaluating the ability of the distributed system to achieve its goals when presented with a set of operational challenges. This approach allowed design alternatives to be explored, allowed trade-offs to be exposed, provided insight into the parameters that influence overall system performance, and facilitated the identification of requirements for further development.

Parts of this effort were sponsored by the Department of the Navy, Office of Naval Research. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the Office of Naval Research.

TABLE OF CONTENTS

I. INTRODUCTION

General Objectives and Specific Aims

The Navy/DoD is moving towards deployment and operation of large-scale networks of autonomous and semi-autonomous sensors, platforms, and weapons that will improve the nation's ability to face a variety of threats. The Office of Naval Research (ONR) established the Deployable Autonomous Distributed System (DADS) program to demonstrate the feasibility of an advanced tactical/surveillance system that operates as an autonomous field of underwater distributed sensor nodes using a cooperative field level detection and data fusion system. The present DADS concept utilizes centralized data fusion, target tracking, target classification, and control functions under control of a master node. The organization of future DADS (called Micro-DADS) will use a new architecture of distributed clusters of sensors to allow for improved performance. Since the master node is eliminated in the new architecture, the system functions are required to be distributed among the clusters in Micro-DADS, allowing for increased autonomy in some modes of operation. The increased autonomy of Micro-DADS will make it possible to employ new methods to reduce vulnerability to detection and destruction by the enemy, but presents additional challenges including detecting, tracking, and classifying targets in a distributed environment, coordination of autonomous clusters and prolonging the lifetime of the battery-powered sensor field. The general objective of this research was to investigate key distributed coordination and control technologies needed for Micro-DADS. The specific tasks accomplished during the course of the research include:

1) Identification and exploration of alternative

   a) Organizational and computational hierarchies for a distributed intelligent system architecture that support the ability of the system to achieve directed goals by performing complex tasks in a distributed environment.

   b) Methods for distributed data fusion, control, and coordination to allow the system to operate in modes ranging from those requiring a high degree of cluster autonomy to modes requiring high bandwidth communication and coherent processing.

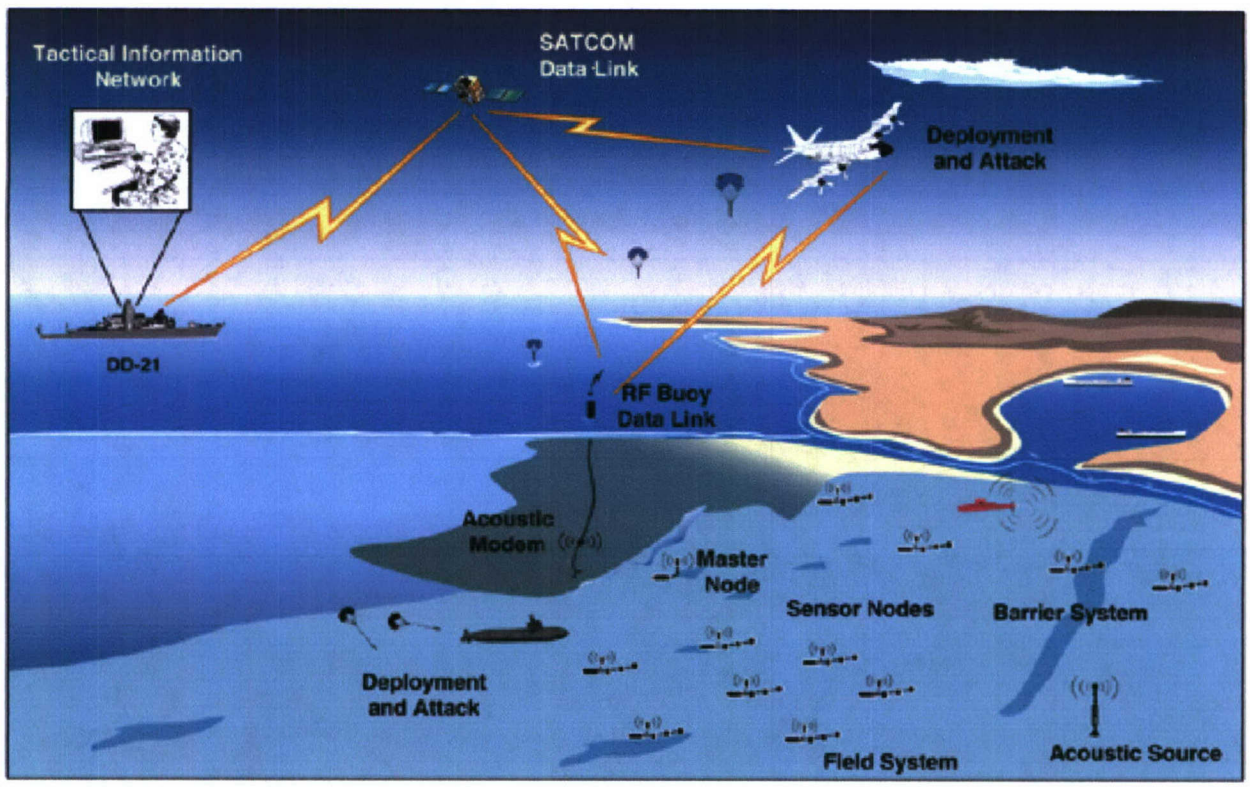
   c) Strategies to allow the sensor field to survive attacks, failures, and accidents.

2) Development of new intelligent software agents to facilitate the exchange of information, distribution of the data fusion, control, and coordination functions, support achieving survivability, and provide a high degree of autonomy when needed.

3) Development of computer simulations for evaluation of algorithms and alternative approaches identified in tasks one and two above. Simulations were used to develop and test alternatives, consider tradeoffs, and evaluate the ability of the system to achieve its goals when presented with a set of operational challenges. The use of visualization and virtual reality environments were explored as tools to understand, interpret, evaluate, and compare tradeoffs and efficiency for the complex Micro-DADS.

The Present DADS Concept

The present DADS concept (Fig. 1) is a networked sensor system that includes a field of underwater sensor nodes that communicate via telesonar [1-4]. The DADS is capable of operating in shallow-water environments such as a harbor, beach, or chokepoint. Small *sensor nodes* sit on the ocean floor and may contain acoustic sensors, electric field sensors, and vector magnetometers. Data is collected, processed, and locally fused in the sensor node, which then forwards information about target detections to a *master node.* The master node fuses the sensor outputs and controls the field power usage to maximize system lifetime. The master node sends its data acoustically to a *gateway node*, which communicates with the external command center via RF communications. The nodes run on battery energy and communicate with each other using underwater acoustic modems, in many circumstances relaying messages and data from node to node. At each node, power is consumed by the processing associated with target detections and by the communications used to transmit and relay detections acoustically. Weapon nodes could be integrated into the field if an intelligent minefield is the ultimate objective.



(from http://www.onr.navy.mil/sci_tech/ocean/321_sensing/info_deploy.htm not available March 2007.)

Fig. 1. Conceptual DADS field description. The target to be tracked is the red submarine located at the right of the figure just outside the mouth of the bay (if the figure appears in black and white, the submarine is dark gray).

The development of the DADS concept has presented many unique challenges and opportunities for research and technology development. Technical challenges to be overcome in order to maximize the performance and life of a DADS include energy limitations (battery-powered nodes), difficulties in achieving near real-time signal and fusion processing imposed by the limited computational capabilities of node microprocessors, variability of the environment affecting the consistency of sensor data required to support state estimation by the fusion engine, large gaps between track segments due to the configuration and density of the sensor field, the need to control the reporting of false alarms to reduce waste of resources, the need for an automated target classification capability, and heterogeneous multisensor detections that make it difficult to correlate tracks and sensor attributes.

The master node carries out central coordination of the major data fusion, control, and communications functions. The following brief review of the sensing and target detection, data fusion, and control functions in the present centralized DADS provides the background to understand the research needed to develop the future Micro-DADS.

Sensing and Target Detection

Each sensor node uses a set of acoustic and electromagnetic sensors to provide coverage of a relatively small area of interest. Intra-node fusion (with cross-cueing between acoustic and electromagnetic sensors) performed at the sensor nodes significantly reduces the data transmission from each sensor node and offloads some of the fusion processing at the master node. Both the acoustic and magnetic sensors must perceive a target at the sensor node to report a detection. Once one node has detected the target, a second node nearby is cued and another sensor node must detect the target. Once the second sensor node detects and reports the target, a field level detection is called and reported out by the master node for field level fusion. This processing results in the reporting of tracklets that provide high confidence position, course, speed, and classification attributes. This approach provides strongly correlated sensor reports, reducing the number of uncorrelated or weakly correlated detections that occur at the sensor nodes.

Data Fusion for Target Tracking and Classification

Correlation Methods and Strategies. Given the wide variety of deployment areas in which DADS must operate, it must be robust against many factors, including environmental concerns and field configurations [1], [2]. Field configurations consist of the spacing between the sensor nodes in the field and the layout of the sensor field. In a dense field where sensor coverage overlaps, textbook correlation algorithms can be employed. However, in the sparse DADS field, overlapping sensor node coverage is minimal or nonexistent, and the lack of data due to long periods between detection reports becomes a significant issue. The correlation of data and tracks from nonhomogeneous sensor types reporting different target attributes also requires a careful application of correlation methods. A Multiple Hypothesis Tracker Correlator (MHTC) that is robust in correlation and tracking problems performs the field-level data fusion at the DADS master node. In the MHTC concept, hypotheses are formed based on the association and correlation of sensor reports. Each hypothesis consists of a different combination of sensor reports, an association confidence, and a tracking confidence. Correlation processes can be vastly improved by using *in situ* environmental information as well as information external to the field.

Target kinematic, target attribute, and environmental measurements are the prime inputs. At the field level (master node), knowledge about the environment can be exploited to control processing at the nodes and provide the best opportunities for correlation of sensor detections. Likewise, information about likely targets in the area can be used to adjust correlation confidences. The MHTC allows soft decisions to be made until more data are received. Drawbacks entail the use of more memory due to potentially large combinatorics and the addition of pruning rules to manage the hypotheses. Given the variability in the lay down and the spacing of the sensor nodes, and other factors such as target density, a statically tuned MHTC would not yield the performance needed by the DADS program. For DADS, an adaptation capability has been added to allow the MHTC to be robust across all possible field configurations and environments. In a study undertaken to assess the benefits of maintaining large numbers of hypotheses when operating in a DADS-like environment, a single hypothesis approach (nearest neighbor tracker) had poor performance for sparse field configurations but a limited hypothesis tracker (3 hypotheses) exhibited a performance comparable to the full MHTC [1,2]. Reduced performance in cases where limiting the tracker to three hypotheses prompted the development of adaptive methods for pruning hypotheses. Fuzzy control has been studied as means to provide efficient hypothesis management in MHTC [3].

Classification. Automatic target classification is important in DADS. Since there is no human operator in the autonomous sensor system, there is a need to reduce false alarm reporting from the field, and there is a need to address the levels of refinement in target classification and their uncertainties. The primary target classification process takes place in the DADS master node. The automated classification requires a) databases of sensor attributes and target characteristics and b) a process to combine the received information to produce a classification estimate. Several projects have extended methods originally developed for the classification of subsurface targets based solely on acoustic data for additional targets and data inputs. A fuzzy conditioned Dempster-Shafer algorithm (FCDS) has been developed to determine the classification estimate. FCDS is a fully probabilistic theory, consistent with Bayesian theory, which is appropriate for reasoning with ambiguous and imprecise evidence [2]. FCDS is capable of incorporating *a priori* knowledge of targets.

Control for Optimizing Performance and Life

Both the processing of sensor detections and acoustic communication drain a node's battery. In order to maximize system performance and lifetime, the master node controls the field power usage by a) adjusting detector thresholds and b) the routing of acoustic communications. Given the sensor locations, communication cost between nodes, initial power available at each node, power consumption of the sensors, processing and communication, the primary DADS control problem is to adjust sensor thresholds and communication paths between nodes to maximize field life subject to a constraint of meeting a desired field level probability of detection.

An individual sensor node can detect targets over only a small area of coverage. To reduce sensor detection "holes" in sparsely spaced fields, the master node controls the field by adjusting the sensor node thresholds to acquire a target of interest and detect it through the field [4]. The master node cues the field by directing selected sensors at the nodes which are near the target to reduce their detection threshold, thereby increasing probability of detection and hence the sensor area of coverage. Since reducing thresholds also results in an increase in false alarms and

potentially an increase in communications and power drain, reducing all of the sensor node thresholds would limit the system operation and therefore is not acceptable. At individual sensor nodes, thresholds are lowered or raised to maintain the desired constant field level probability of detection, while maximizing the life of the field. Threshold levels are adjusted at the sensor suite by choosing different operating points on a receiver operating characteristic (ROC) curve to yield different probabilities of detection and probabilities of false alarm.

Evolutionary programming methods have been applied in DADS to allow the field to be controlled by the master node to meet a field-level probability constraint (via threshold adaptation) and to optimize routing of the sensor node message traffic at minimal power cost, doubling the life of the field [5]. Messages were routed through alternative paths excluding nodes with low battery reserves if possible; this scheme allowed power to be conserved where needed in order to keep nodes from being lost due to exhaustion of power reserves. Optimization was based on a cost function for a detector constructed to represent the estimated power consumed over a period of time T at each node n, $n=1, \ldots N$. The cost function model accounted for communication costs, probabilities of detection and false alarm, node spacing of the field, and signal processing parameters used at the sensor node.

Future DADS Concept (Micro-DADS)

Future DADS (referred to as Micro-DADS) are expected to be organized in multiple clusters typically consisting of 6 to 12 sensor nodes plus a cluster node (Fig. 2) [6]. Micro-DADS sensor nodes may include new sensing technologies in addition to the acoustic, electric field, and magnetic sensors in current DADS. The cluster node communicates with its sensor nodes and may perform signal processing and data fusion. The cluster may include decoy nodes and nodes that may be relocatable. The cluster node is used in routing communications between clusters and other entities in the field. An RF link or satellite will allow communication with an external command center that may specify commands, configuration, operating modes, priorities, and scenario knowledge. Micro-DADS may be armed with internal or external lethal or non-lethal devices.

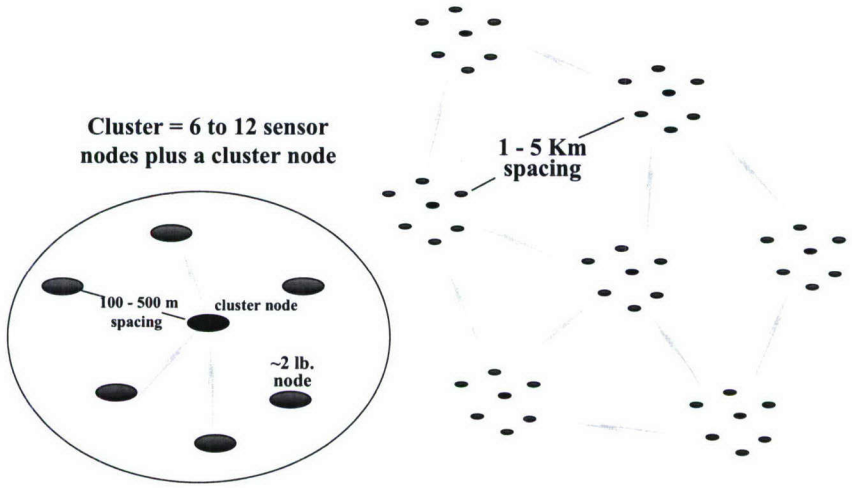


Fig. 2. Conceptual Micro-DADS Field Description (from the Future DADS and USW Concept Briefing, Office of Naval Research Arlington, VA, May 29, 2001 [6]).

Clustering the sensors, new sensor and packaging technologies and distributed data fusion, control, and coordination in Micro-DADS are expected to make the network less vulnerable to detection, dredging, trawling, and such future threats as quiet submarines, autonomous and unmanned underwater vehicles, and high-speed torpedoes. The autonomy of the distributed Micro-DADS clusters will offer advantages not realized with the centralized management scheme employed in the current DADS. Scalability and fault tolerance are key attributes of increased autonomy that should increase the capability of the system to withstand attack and reconfigure if sensors or clusters are lost.

The clustering of sensors provides several advantages that may allow the sensor coverage areas to overlap, avoiding some of the challenges presented by the sparse DADS field. Clustering the sensors may allow coherent processing concepts and high bandwidth communications to be employed within a cluster to improve detection, tracking, and classification performance in some modes of operation. In addition, some clusters may include specialized sensors that may carry out unique functions. If the sensors within a cluster have significant processing power, then it may be possible to employ distributed processing methods to increase the available computing power above that of the cluster node alone.

Local data fusion within a cluster would significantly reduce (or if in some operating modes there is a high degree of autonomy, perhaps obviate) the transmission of low-level data from each cluster node and reduce or eliminate the need for data fusion outside the cluster. Fusion within the cluster may make use of cross-cueing between differing or similar sensor technologies to provide strongly correlated sensor reports, reducing the number of uncorrelated or weakly correlated detections that occur in the cluster. Cueing may also be employed within a cluster to allow individual sensors or groups to track while lowering thresholds of other sensors or shutting them off completely to save power. As a target moves out of range of a cluster, inter-cluster cueing may be needed to allow neighboring clusters to continue tracking.

Multiple and perhaps redundant sensors within a cluster node may allow the application of new methods for conserving power. Power would be saved if sensors could be turned on only when needed to acquire measurements.

Autonomous Distributed Systems

The autonomy of the distributed Micro-DADS clusters will offer advantages not realized with the current centralized DADS. An autonomous distributed system (ADS) is a collection of independent entities that interact with one another to accomplish a given task. The properties of autonomous action and autonomous interaction enable an ADS to possess properties like on-line maintainability, reconfigurability, and fault tolerance. In multisensor surveillance systems, the autonomous action and interaction must address the data fusion problem and the coordination of the deployed sensors. Data fusion defines the optimal way of combining detections, local tracks, and attributes received from several sensors before presenting the information. Decentralized systems offer the potential benefit of parallel processing in data fusion. The coordination problem addresses the optimization of the local sensor data acquisition and processing considering low-level local sensor information, the results of the data fusion process, and high-level information from other sensors or nodes.

The literature on synthesizing decentralized decision-making algorithms and evaluating their performance is sparse. Since centralized control gathers all of the necessary data from all entities in the system and utilizes the data to make decisions, there is often a belief that the centralized decisions may be "globally" optimal and superior to the performance of decentralized systems. However, the combination of a) latency of the information received from subordinates, b) the delay in making decisions that results from the significant computational load, and c) the latency of transmission of the decisions hinder the generation of high-quality decisions with centralized control, especially as the frequency of arrival of the sensor data increases and the environment gets larger, dynamic, and more complex. In contrast, while the decentralized paradigm successfully addresses issues b) and c), the lack of access to the total system state limits the quality of its decisions. Recent literature includes successful asynchronous, distributed, decision making algorithm designs, wherein the local decision-making at every site replaces the centralized decision making to achieve faster response, higher reliability, and greater accuracy of the decisions [7]. Based on a distributed hybrid control paradigm, MFAD is a mathematical framework for asynchronous, distributed systems that permits the description of centralized decision-making algorithms and facilities the synthesis of distributed decision-making algorithms. MFAD was employed to develop a distributed systems approach for a simulated military command, control, and communication problem. In a comparative analysis, decentralized decisions derived while trying to achieve globally optimum behavior were superior to those of a centralized approach. The quality of decisions made by the decentralized system was evaluated by comparing the decentralized decisions to decisions that are considered ideal because they are made utilizing complete knowledge of the total system state, unlike the decentralized paradigm, and are not subject to the latency inherent in the centralized paradigm. The decentralized decisions closely tracked the ideal decisions that, though unattainable, provide a fundamental and absolute basis for comparing the quality of decisions.

New Challenges in Micro-DADS

Although new technologies such as self-contained power, fuel regeneration, and microelectromechanical sensors may offer significant future improvements in some areas, Micro-DADS faces many of the same technological limitations as the current DADS. Energy limitations are expected to continue to have a major impact on field life and performance given the power consumption of sensing, signal processing and communication activities. Technological limitations, power considerations, and the need to control the rate of false alarms make maximizing field life subject to a constraint of meeting desired field level probabilities of detection and false alarm a major component of the control objective for the new Micro-DADS. However, the cluster architecture and an emphasis on reducing field vulnerability in Micro-DADS require new distributed data fusion, control, and coordination approaches that may drastically change the communications and sensor management. The centralized data fusion, tracking, target classification, and control performed by the master node in the current DADS concept must be replaced by new distributed methods appropriate for the new cluster architecture, advanced features, and new goals of the Micro-DADS. Hence, there are new trade-offs to be made between the degree of cooperation and information exchange between clusters, the degree of autonomy, performance, robustness, field life, and vulnerability to detection in the distributed Micro-DADS architecture.

Interactions with ONR personnel shaped the research and provided a broad background perspective. Mr. Oliver Allen, Program Officer at the time of the grant award, and Mr. Larry Green visited Dr. Jannett at UAB on August 4, 2003 for a kick-off meeting at which the grant activities and plan were reviewed. At that time, in addition to the originally-proposed work that considered relatively sparse networks of sensors that reported range and bearing, Mr. Allen requested that the project be adjusted to consider a sensor dust concept employing a large number of relatively inexpensive and unintelligent sensors, perhaps having limited communication and detection capabilities such that the sensors are only able to indicate the presence or absence of a target. Mr. Robert Wingo replaced Mr. Allen as Program Officer in January 2004. A briefing on the status of the work was given by Dr. Jannett at the Survivable Undersea Sensors (SUS) Workshop that was organized by Mr. Wingo and held in Arlington, VA on April 23, 2004.

The rest of this report is organized as follows. Section II describes the organizational and computational hierarchies selected to provide an intelligent architecture for the distributed system. Section III outlines strategies that can be employed to improve the survivability of the distributed sensor field. Section IV describes the algorithms employed for data fusion, control, and coordination in the distributed sensor network. Section V describes the use of intelligent agents in implementing the distributed data fusion, control, and coordination algorithms. Section VI describes the simulation studies used to test and evaluate architectures, survivability strategies, and algorithms. Section VII presents conclusions. A technical report and three manuscripts of recently submitted papers offering detailed descriptions of key parts of the research are given in the Appendix. Papers published by the research group describing details of the research and results are referenced throughout this report.

II. ORGANIZATIONAL AND COMPUTATIONAL HIERARCHIES FOR A DISTRIBUTED INTELLIGENT SYSTEM ARCHITECTURE

In general, centralized management schemes for data fusion and coordination require data and decisions to be communicated to or from the central entity. Weaknesses of centralized methods include a high vulnerability to failure and a lack of scalability [7]. If the central entity or any of its key components fails, the system cannot function. As the system complexity and size increase, the system becomes limited by the processing and communication capacities of the central manager.

In decentralized approaches, the decision making process is distributed among several network nodes. Every entity utilizes its local information and local goals, along with appropriate coordination information obtained from other entities, to make its decisions autonomously, but cooperatively, in order to achieve the desired global performance. Compared to the centralized system, the distributed decision-making system can process information faster and can react more quickly to dynamic changes in the environment. The distributed system is expected to exhibit scalability and robustness against catastrophic failure. However, if the advantages of decentralized realizations are to be attained, resources and functionality must be replicated across the distributed system at an extra cost. Since each entity computes decisions autonomously, extra communication, data storage, and computational overheads are incurred in making key system information, including the global goals, available to each entity. Distributed systems realizations may require extra hardware resources if their advantages over centralized systems are to be fully realized. For example, if only one node entity has the ability to communicate with the command center in a centralized sensor network, then a distributed realization of the sensor network would require that multiple entities have the ability to communicate with the command center. Otherwise, the loss of the single entity that communicates with the command center would render the system useless.

Organizational Hierarchies

At a minimum, intelligence involves sensing the environment, making decisions to achieve a goal, and controlling action [8]. Intelligent systems generally exhibit a hierarchical organization, include certain elements, and require an interconnecting system architecture for these elements. Several factors have been identified that affect the degree of intelligence, including sophistication of algorithms, quality of information, computational power, the values used to make decisions to choose among alternatives, and communication capability. In this work, a distributed intelligent system architecture of organizational and computational hierarchies that is consistent with characteristics of intelligent systems was employed to support the ability of Micro-DADS to achieve specified goals by performing complex tasks in a distributed environment. An organizational framework structures the interactions between individuals and affords individual entities an abstract view of the task accomplishment activity going on in the system. An organization imposes roles, expectations, and relationships among entities. Organizational structures facilitate task decomposition and allocation, resource sharing and problem solving coherence.

Fig. 2 includes sensor nodes and cluster nodes, but does not show master nodes or the external command center that receives the data from the sensor network. Other hardware, such as

the communication link(s) between the network and the external command center, is also not shown.

The organizational hierarchy employed in this work includes the sensor, cluster, and field levels, with the external command center located at the highest organizational level. The sensors are organized into clusters, such that each cluster contains a fraction of the field's sensors. The external command center communicates with the field about the status and activities of the field. The networks considered in this work include a) sensor, cluster, and master nodes, or b) sensor and cluster nodes. Sensor, cluster, and master nodes map directly into the sensor, cluster, and field levels, respectively, of the organizational hierarchy. If no master nodes are present in the system, then the cluster nodes also map into the field level of the organizational hierarchy.

This organizational hierarchy is consistent with the partitioning of spatial and temporal resolution that characterizes intelligent systems [8]. Low-level data is processed at the sensor level, with higher organizational levels operating on increasingly more abstract data. Sensor measurements reflect the status within a small subset of the field, and larger portions of the field are considered at the cluster and field levels.

Organizational hierarchies may configure groups of sensors in fixed clusters such that the sensors within a cluster report to a single cluster node. Instead of this fixed clustering approach, some applications may be better served by a dynamic clustering approach in which the sensors are organized into the clusters that best facilitate the application [9-12]. For example, in a target detection application using fixed clustering, if the target straddles two clusters, then the sensors nearest the target within each cluster would detect, and data from both clusters would need to be processed in making a detection decision. Performance might be improved with use of a dynamic clustering approach in which a single cluster is formed to include the all of the detecting sensors. Fixed clustering was used in this research in order to facilitate the development, demonstration, and ease of comparison of distributed systems methods, and approaches to improve survivability. However, the research results may find application in dynamic clustering as well as fixed clustering.

Computational Hierarchies

Many different algorithms must be executed in order to carry out the various sensor network functions. The computational hierarchies depend on the specific tasks that must be accomplished to execute the algorithms. A methodology for the design of intelligent systems that describes an architecture, design guidelines for computational hierarchies, building blocks, and a prototyping method was employed in this work [13]. This methodology resulted in a system that exhibits the characteristics that define intelligent systems, including a hierarchical organization, and the use of multiresolutional information processing in external information organization, knowledge representation, and decision-making processes. The following guidelines were used to facilitate the design of intelligent sensor networks.

- Use task-oriented decomposition to design the computational hierarchy for each algorithm. The steps are developing a task tree, choosing a thread of tasks spanning the tree, and adding task threads iteratively.

- Map the task tree into the organizational hierarchy of nodes, which may be thought of as intelligence modules.
- Organize the hierarchy around tasks top down and equipment (actuators and sensors) bottom up.
- Partition spatial and temporal resolution by an order of magnitude between levels in the hierarchy, with roughly ten decisions or less per level.
- Use 7 ± 2 subordinate nodes per supervisory node and one supervisory node at a time.
- Distribute sensory processing, world model, behavior generation, and value judgment functions throughout the system so that they exist in appropriate forms at each node.

In this work, the computational hierarchy maps the tasks needed to execute a given algorithm into the sensor level functions, cluster level functions, and master level functions needed to execute the algorithm. Master functions represent field-level functions. The term 'master' was inspired by the previous centralized DADS work in which the master node carried out field-level functions. As described earlier, the networks studied in this work included a) sensor, cluster, and master nodes, or b) sensor and cluster nodes. Sensor functions were carried out at the sensor nodes and cluster functions were carried out at the cluster nodes. Master level functions must be carried out using the nodes that exist in a given physical realization. If master nodes were included in the network, then master level functions were carried out at master nodes. If no master nodes were included, then master level functions were carried out at cluster nodes.

Motivated by the *Principle of Increasing Precision with Decreasing Intelligence*, a three-level computational structure which includes an organization level, a task planning level, and a task execution level was replicated at each of the sensor, cluster, and master levels [14]. The organization level receives external commands and information, communicates with other nodes, and performs such operations as planning and high-level decision-making and sends commands to the task planning level. The task planning level receives the decisions about the set of tasks to perform in the next sensor management cycle and their priority. The execution level is concerned with the low-level sensor management, data acquisition and processing.

III. STRATEGIES TO ALLOW THE SENSOR FIELD TO SURVIVE ATTACKS, FAILURES, AND ACCIDENTS

Survivability addresses the ability of a system to respond to attacks, failures, and accidents [15]. Survivability depends on four key properties: a) resistance to threats, b) recognition of threats and the extent of any damage suffered, c) recovery of full and essential services after damage, and d) adaptation and evolution to reduce effectiveness of future threats. The general trend in designing distributed systems to give an increasing amount of autonomy to the individual nodes generally increases survivability [16]. The distributed architecture of Micro-DADS inherently offers physical dispersion and the capacity for functional distribution that support the resistance and recovery properties. Highly autonomous operating modes may facilitate reduced communications that decreases the likelihood of detection by the enemy and thereby increases the resistance to attack.

The coordination and communication network functions are major determinants of survivability. These and other Micro-DADS system functions were considered in addressing the four survivability properties. The major threat considered in this work was the loss of a node due to depletion of power reserves or node failure. In this work, the network lifetime was assumed to end with the loss of any sensor node, cluster node, or master node. Although a different definition of the useful network lifetime may be more appropriate in some network applications, the definition above serves as a very direct reflection of the major problems that limit network lifetime: loss of nodes due to node failure and depletion of battery reserves. Threats due to factors such as communications disrupted by the enemy, interception of communications by the enemy, or false communications injected by the enemy were not considered.

The primary strategies to increase survivability developed and exploited in this work include 1) incorporating and utilizing redundancy 2) balancing resource utilization among available reserves, and 3) using local optimization to improve global performance. These strategies support the four key survivability properties. Incorporating and utilizing redundancy is a strategy that is appropriate to use when the nominal field layout does not include the active resources needed to recover full and essential services after damage. In this strategy, redundant nodes distributed within the field are dormant until they need to be utilized to achieve a higher level of performance or to replace nodes lost due to failure or depletion of power reserves. In the strategy of balancing resource utilization among available reserves, functions are switched between node entities to evenly spread the utilization of computational and communications resources over the network so that nodes are not lost prematurely due to exhaustion of battery energy. Without balanced resource utilization, some nodes will die to end the useful lifetime of the network even though other nodes may have enough battery energy.to allow the network to run significantly longer. The strategy of using local optimization to improve global performance allows global performance to be optimized without requiring extensive global information sharing and centralized decision-making. These three approaches were evaluated using agents and without using agents as methods for managing the coordination and communication functions to support the resistance, recognition, recovery, and evolution survivability properties.

IV. ALGORITHMS FOR DISTRIBUTED DATA FUSION, CONTROL, AND COORDINATION

For Micro-DADS, significant efforts that require resources far beyond those available for this work will be required for the full and detailed development of distributed algorithms needed for sensing and target detection, data fusion, control, coordination and communications functions that are as sophisticated as those that have been developed for the centralized DADS. The intelligent systems design methodology described above in the section **Computational Hierarchies** allows relatively limited aspects of a problem to be addressed, with possible expansion to other aspects at a later date. This capability was very important since it allowed us to assess aggregate system performance by evaluating a number of alternatives for algorithms without the detailed development of all of the algorithms that would eventually reside in an entity (sensor node, cluster node or master node). Although the following descriptions of the major Micro-DADS system functions are presented in some detail, our primary efforts were focused on the development of the control and coordination functions that are so important in distributed systems. Thus, instead of developing each of the functions in great detail, the primary approach used in this work was to model the behavior of candidate algorithms to allow the assessment of how their performance would influence the system. In situations where a detailed implementation of an algorithm for a sensing and target detection, data fusion, control or coordination function was not available, high-level abstract representations of function behavior were used to facilitate constructing the system-level simulation. This approach provided insight into the parameters that influence overall system performance, exposed trade-offs, allowed design alternatives to be explored, and facilitated the identification of requirements.

For Micro-DADS, the primary system functions are sensing and target detection, data fusion for target tracking and classification, control for optimizing performance and life, and coordination and communications appropriate for the distributed architecture. These functions will be carried out by operations within the clusters, and by the clusters cooperating within the Micro-DADS architecture. The new opportunities and options offered by clustering emphasized the need to develop appropriate methods for distributed data fusion and coordination as described in the following.

Sensing and Target Detection

Determining the presence or absence of a target within the sensor field is the detection problem. Determining the position of the target within the field is the localization problem. Determining where the target is moving is the tracking problem. Distributed detection, localization, and tracking were considered in this work. In one approach, many inexpensive sensors that reported a logic '1' or '0' to report the presence or absence of a target, respectively, were employed. The *a priori* knowledge of the positions of the detecting sensors were used to localize the position of the target and report it in (x, y) coordinates. In another approach, sensors that reported the target's range and bearing were used in a target tracking application.

Target Detection Using Binary Sensor Data in Hierarchical Sensor Networks

Sensor networks employing a large number of relatively inexpensive sensors having a limited detection range and communication capability are an area of significant research interest

[17], [18]. For a sensor network having a large number of sensors, a fusion center can make a final decision about a target's presence with a decision fusion rule that uses the total number of detections reported by local sensors as a statistic in binary hypothesis testing. However, sensor limitations may make it difficult to achieve a required probability of detection (PD) and probability of false alarm (PF) performance unless the number of sensors is very large [17]. In addition, this centralized approach would suffer from weaknesses including a high vulnerability to failure and a lack of scalability. As the size of such a centralized system increases, its performance is limited by the computational and communications capacities of the central entity.

We investigated the benefits offered by multi-tier hierarchical sensor networks that utilize decentralized fusion for target detection (see appendix). The PD and PF that describe the detection performance at the different levels of a multi-tier network were derived and calculated. We devised and demonstrated a strategy in which detection decisions are made through repeated trials in order to facilitate achievement of a specified average PD and PF performance using inexpensive sensors that have limited detection and communications capabilities. The multi-tier network allowed repeated trials to be performed at the different hierarchical levels in order to obtain the extra degrees of freedom that are needed to achieve the specified average PD and PF performance. In addition, if the low-level detections are made within a sensor cluster whose location within the sensor field is known, then the location of the detecting cluster is information that could aid in target localization.

This work may facilitate a variety of applications for networks that utilize inexpensive sensors. For example, inexpensive sensors might be arranged to form the outside border of a sensor network such that a high PD and low PF are achieved within the border area. When a target is detected at the border, other network functions or application-specific sensors could be awakened and used as needed to estimate the target's position, track the target, or classify the target.

The distributed nature of multi-tier hierarchical sensor networks facilitates decentralized fusion and performance improvement using repeated trials. Decentralized fusion reduces the communication resource demand at areas within the network that are near the central entity. However, the use of repeated trials requires increased sensing and local communications. Future work could consider optimizing performance and resource utilization by the choices of parameters such as the type and the number of common sensors, the number of sensors at each level, detection thresholds, and the number of hierarchical levels. Another area for future work is the development of methods that would guarantee that the worst-case performance exceeds a specified level.

Maximum Likelihood Target Localization Using Binary Sensor Data

Target localization for estimation of target position is an important application of wireless sensor networks (WSNs). Position estimates obtained using localization can also be used in tracking applications. One promising approach for target localization is a method using binary sensor data for which a maximum likelihood (ML) estimator and its Cramer-Rao lower bound have been derived [19]. In this approach, each sensor makes a binary decision about a target's presence by comparing the measured signal strength to a threshold, and communicates a one-bit message to a fusion center. The fusion center uses the binary information received from all the

sensors, along with *a priori* information about the positions of the sensors, to localize the target through nonlinear optimization of a highly complex multimodal function. Recently, this approach was extended for localization based on quantized data [20]. These methods are attractive because they facilitate accurate target localization based on the transmission of binary or multibit quantized data, which requires limited communication bandwidth. The ML estimation framework for target localization using binary data presented in [19] and reviewed in the following was utilized in this work.

In the model below, signal intensity is assumed to attenuate as the distance from the target to a sensor increases

$$a_i = \sqrt{\frac{P_o}{1+\alpha d_i^{\,n}}} \tag{1}$$

where a_i is the signal amplitude at the ith sensor, d_i is the distance from the target to the ith sensor, $\sqrt{P_o}$ is the signal amplitude at the ith sensor when d_i is zero, α is a constant, and n is the signal energy decay exponent. Sensor parameters used in this work are $n = 2$, $\alpha = 2$, and $P_o = 64$. The distance from the target to the ith sensor is given by

$$d_i = \sqrt{(x_i - x_t)^2 + (y_i - y_t)^2} \tag{2}$$

where (x_i, y_i) are the coordinates of the ith sensor and the target location is given by (x_t, y_t). The signal a_i is corrupted by standard Gaussian noise ω_{ij} that is independent across sensor i and time frame j

$$s_{ij} = a_i + \omega_{ij}. \tag{3}$$

Each sensor makes a binary decision depending on the signal s_{ij} satisfying a local sensor threshold η_{ij} during timeframe j and transmits its decision to a higher-level fusion node where the decision is used in target localization. After collecting the decisions I_{ij} from all N sensors for all T timeframes, the fusion center estimates the parameter vector, $\theta = [x_t\ y_t]$, by maximizing a log likelihood function with respect to θ [19]. The elements of θ represent the target position.

For a target position, θ, the Cramer Rao Lower Bound (CRLB) for an unbiased ML estimate of the target position, $\hat{\theta}$, is given by

$$E\left\{\left[\hat{\theta}(I)-\theta\right]\left[\hat{\theta}(I)-\theta\right]^T\right\} \geq J^{-1} \tag{4}$$

where J is the Fisher Information Matrix (FIM) [19]. The covariances of the errors in estimates of (x_t, y_t) are bounded by the (1, 1) and (2, 2) elements in the J^{-1} matrix, respectively

$$\begin{aligned} var(\hat{\theta}_1) = var(\hat{x}_t) \geq J_{11}^{-1} \\ var(\hat{\theta}_2) = var(\hat{y}_t) \geq J_{22}^{-1}. \end{aligned} \tag{5}$$

A lower bound on the variance of the target position estimate, $\hat{D}$, can be computed as

$$var(\hat{D}) = var(\hat{\theta}_1) + var(\hat{\theta}_2) \tag{6}$$

where $D = \sqrt{x_t^2 + y_t^2}$ is the Frobenius norm. Thus, the lower bound for the root-mean square (RMS) error in the overall target position estimate is $\sqrt{var(\hat{D})}$. Computing the lower bound for the RMS error at many target locations across the sensor field aids in assessing the performance of alternative network configurations.

However, the parameter estimation required by this approach involves optimization of a complex multimodal function, which is challenging. Deterministic search algorithms are ineffective because they can converge to local minima, resulting in large localization errors. Stochastic algorithms provide global solutions, but may yield only approximate solutions, are computationally burdensome, and can require tuning of parameters. Although the approach presented in [19]-[20] is attractive and is supported by estimation theory that would facilitate many further developments, it quickly became clear in this research that applications were not possible without the development of a suitable estimation algorithm. Several approaches to this estimation problem were considered in this research [21]-[25]. First, we considered the continuous adaptive culture model (CACM) as an alternative to stochastic estimation algorithms such as Genetic Algorithms (GAs). We also studied a new estimation algorithm, the gradient-based particle swarm optimization (GPSO) algorithm, that combines stochastic and deterministic schemes to achieve high convergence rates and avoid traps due to local minima experienced in ML target localization. Next, work turned to parallel PSO algorithms that may be suitable for complex optimization problems. Finally, we investigated a novel two-step approach that allows accurate localization of the target without incurring a high computational burden.

The CACM Algorithm. GAs suffer from slow convergence rates, and require the user to make difficult choices of ranking and scaling schemes and subpopulations that lead to complexities in implementation. A new computationally inexpensive alternative to GAs, the Continuous Adaptive Culture Model (CACM) algorithm, was proposed [22]. This new optimization algorithm was inspired by sociological models of culture dissemination and uses operators that act directly on vectors of real numbers to avoid the computation associated with binary encoding and decoding in GAs. The new algorithm does not use global information sharing, which makes it amenable to parallel implementation since computational bottlenecks can be avoided. The new optimization algorithm was tested using the De Jong test suite of optimization problems. Simulations were used to investigate the effects of various parameters on the performance of the algorithm. For a four-dimensional Rastrigin test function having multiple local minima, the CACM algorithm converged faster than a GA using 40-bit accuracy per variable and a single population. The CACM algorithm does not require global information sharing, thus avoiding computational bottlenecks, and facilitating parallel implementation. The operators introduced can also be used in GAs to avoid binary encoding.

The GPSO Algorithm. Stochastic global optimization algorithms like GAs, PSO algorithms, and simulated annealing avoid traps due to local minima. However, since stochastic optimization schemes perform a random search of the solution space, they suffer from slow convergence rates,

high computational costs, and poor accuracy of the final solution. Thus, hybrid optimization schemes that combine both stochastic and deterministic schemes to achieve high convergence rates and avoid local traps are of interest. The GPSO algorithm combines a PSO algorithm used for global exploration with a gradient-based scheme used for accurate local exploration [23]. The PSO algorithm is used to go near the vicinity of a good local minimum and then the gradient-based local search scheme is used to find the local minimum accurately. Next, this accurately computed local minimum is used as the global best solution by the PSO algorithm to identify still better local minima, and the cycle is repeated. Thus, with the GPSO algorithm, the global minimum is located through a process of finding progressively better local minima. The phenomenon of failure of ML target position estimation in WSNs due to multimodality of the likelihood surface was explored. High sensor power levels resulted in complex likelihood surfaces. Simulation results show that a deterministic algorithm sometimes converged to local minima, especially when the target was near the edge of the sensor field. The performance of the deterministic algorithm was compared with that of the GPSO algorithm for ML estimation. The use of the GPSO algorithm resulted in efficient global optimization and significantly higher estimation accuracy in a variety of cases. However, the GPSO is a very complex algorithm that presents a high computational burden.

Parallelization of the PSO algorithm. A parallel implementation was developed for the coarse-grained parallelization method for the PSO algorithm that was proposed in [21]. In order to aid in the selection of the parameter values that would allow good performance to be achieved using the parallel algorithm, an analytical performance model was developed [24]. The model allows the run-time performance to be predicted for the various parameter options. Results of preliminary simulations were presented to validate the performance model. Next steps in the work are to test parallel PSO methods and the performance model in complex optimization problems.

The Two-Step Approach. In the first step, a coarse estimate is formed by performing a weighted average of the sensor positions that report the presence of the target. The coarse estimate is then used as the initial estimate for a second step in which a deterministic search is performed using the Nelder-Mead simplex method. This two-step process essentially reduces the global search to that requiring a local search near the global minimum, and thus avoids pitfalls due to convergence to local minima. For an exemplary sensor network, our results showed that the deterministic direct search often failed by converging to a local minimum if the starting point was chosen at random. On the other hand, the two-step approach accurately localized the target with a low computational overhead. Simulation results show that the two-step approach is a simple method that can be used to localize a target using binary sensor data in an ML scenario [25] (see appendix). The two-step approach provides an unbiased estimate of the target position, and allows the ML estimator to achieve localization performance near the CRLB. This approach may be a step toward facilitating the routine use of the promising ML target localization methods described in [19] and [20] in which the use of quantized sensor data reduces the needed communication bandwidth.

Effect of Uncertainty in Sensor Positions on the Accuracy of Target Localization. Most research on target location estimation using binary decisions assumes that there is no uncertainty in the positions of the sensors. In reality, many factors may contribute to some measure of sensor

placement uncertainty, resulting in less than ideal sensor positioning. We studied the accuracy of target localization achieved using detection decisions made by sensors whose positions were assumed to be exactly known, even though the detections were actually generated by sensors having uncertain positions [26]. Computer simulations were used to demonstrate that sensor position uncertainty, along with noise and coarseness of the sensor grid, degraded the accuracy of target location estimation. In future work, models could be developed to describe how sensor position uncertainty degrades the accuracy of estimation of target location.

Data Fusion for Target Tracking

Alternative methods of distributed data fusion for target tracking include the possibility of either partially or fully distributed MHTC, methods based on Kalman filtering, and multiple-model schemes. Use of the full complement of sensors and computational power within a cluster increases the power drain and use of resources. Use of a subset of the available sensors within a cluster may allow for longer life with reasonable performance tradeoffs. In addition, the potential benefits of fusing data, tracks, or classifications from multiple clusters needs to be explored and the trade-offs considered; operating modes that support significant field level fusion of data from multiple clusters need to be developed and evaluated if they offer higher performance in some situations. In this work, we have used extended information filtering algorithms as exemplary algorithms because they are applicable for the wide variety of distributed architectures of interest and because they accommodate missing data and late communications [27].

Representation of the Tracking Problem

The instantaneous distance of the target from a sensor can be determined by two parameters, range (ρ), and bearing (θ). Let the velocity of the target be V. The velocity can be resolved into two vectors in x and y directions, V_x, and V_y, respectively (Fig. 3). In a similar fashion, the range ρ of the target from the sensor can be resolved as ρ_x and ρ_y [27].

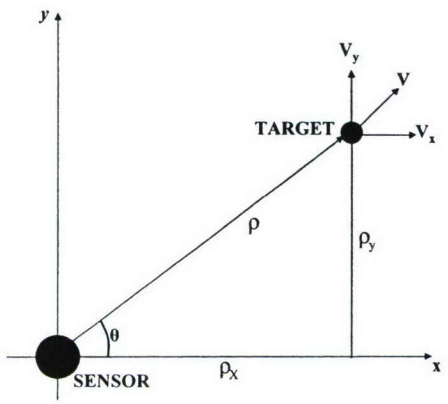


Fig. 3. Tracking problem in two-dimensional space.

If the velocity is assumed constant, the state-space model for the target motion is

$$\begin{bmatrix} \rho_x(k) \\ V_x(k) \\ \rho_y(k) \\ V_y(k) \end{bmatrix} = \begin{bmatrix} 1 & T & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & T \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \rho_x(k-1) \\ V_x(k-1) \\ \rho_y(k-1) \\ V_y(k-1) \end{bmatrix} + w(k) \tag{7}$$

where T is the time interval and $w(k)$ is the process noise model. The measurement model is

$$\begin{bmatrix} \rho \\ \theta \end{bmatrix} = \begin{bmatrix} \sqrt{\rho_x^2 + \rho_y^2} \\ tan^{-1}\left(\frac{\rho_x}{\rho_y}\right) \end{bmatrix} + v(k) \tag{8}$$

where $v(k)$ is the measurement noise model. The representation in (7) and (8) was developed for sensors that report range and bearing. Other representations might be derived for different measurements, such as the estimates of the (x, y) target position provided by localization using binary sensor data [19], [20], [25].

Tracking Algorithms

In pilot studies, we have used extended information filtering (EIF) to apply parallel Kalman filtering algorithms [28]-[31] to the nonlinear tracking problem represented by (7) and (8) above for the two-tier hierarchical, fully connected and non-fully connected distributed architectures of interest. These algorithms accommodate missing data and data that arrives late due to communication delays [27]. These new EIF tracking algorithms serve as exemplary algorithms for distributed data fusion that operate in scenarios ranging from coherent parallel processing at high data rates within a sensor cluster to autonomous tracking based on intermittent and delayed measurements from a single sensor. Hence, the algorithms apply in situations that include tracking a target based on measurements from distant sites that have a single sensor, measurements from sensors within a single cluster, and information from several sensor clusters. The algorithms apply in the following situations:

1) All sensors collocated. This case corresponds to a Micro-DADS cluster tracking a target based on only the measurements from sensors within the cluster. If each sensor node has significant computing power and estimates a local track based on its own measurement, then a parallel EIF structure may be employed to allow the cluster node to produce an optimal global estimate and error covariance based on only the local estimates and covariances that are communicated to it by the local sensor nodes. If each sensor node is not capable of performing the computations needed for local tracking, then standard tracking algorithms may be employed at the cluster node to process the raw sensor data without the benefit of parallel computation.

2) All sensors dispersed. This case corresponds to a situation in which multiple clusters each operate with only a single active sensor. If the measurements at the local sensors are time-sequential in nature, then a second parallel EIF structure may be utilized to achieve

parallelism. Since the algorithms accommodate missing data, the requirement that the measurements be time-sequential is not limiting.

3) Sensors both collocated and dispersed. This case corresponds to generating a global track based on information from several Micro-DADS clusters. A two-tier hierarchical structure is employed to achieve parallelism. At the lowest level, the methods of situation 1 above are used to generate a track estimate for each cluster; a second parallel structure based on the methods of situation 2 above is employed to fuse the estimates from all clusters to produce a global estimate.

The situations above cover the two-tier hierarchical, fully connected, and non-fully connected distributed architectures that represent the spectrum of interconnections relevant for data fusion in the tracking problem. In a two-tier hierarchical architecture, data from each sensor within a cluster is fused at the cluster's local fusion center, and the local fusion results are then fused at a higher-level global fusion center (Fig. 4). In the fully connected (Fig. 5) and the non-fully connected (Fig. 6) architectures, nodes generate estimates based upon both their own local measurements and the measurements from the other nodes to which they are connected.

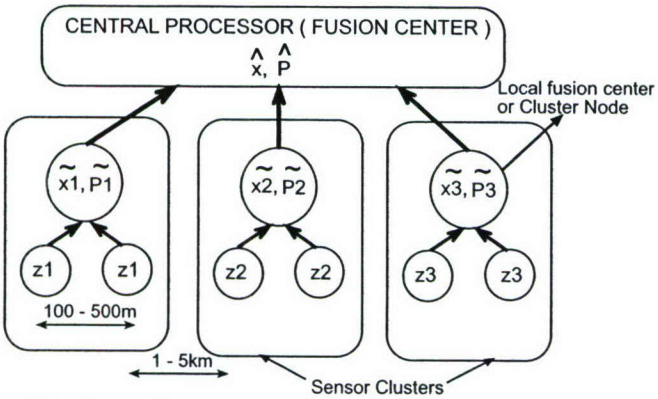


Fig. 4. Two-tier hierarchical architecture. Clusters fuse measurements, z, and pass their local estimates ($\tilde{x}, \tilde{P}$) to a fusion center.

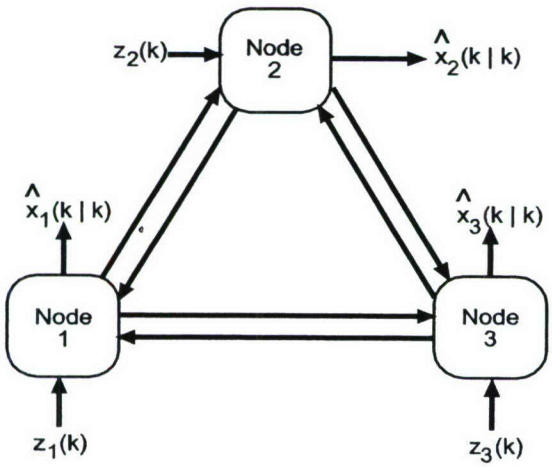


Fig. 5. Decentralized, fully connected architecture. Clusters fuse their measurements, z, and pass their local estimates ($\hat{x}$) to all other clusters.

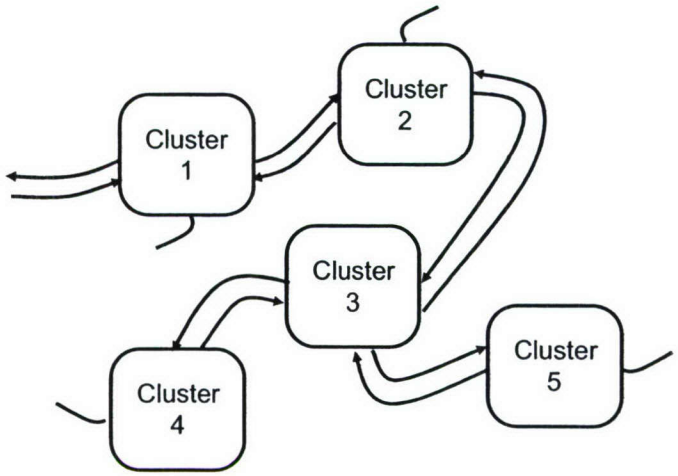


Fig. 6. Distributed and decentralized, non-fully connected architecture. Clusters pass their local estimates to neighboring clusters.

These three architectures may be used to represent interactions between sensors in a cluster or they may represent higher-level interactions between clusters. Any or all of the three architectures may be utilized at different levels within a sensor field. For example, a fully connected architecture may represent the interactions of sensors within a cluster, a two-tier hierarchical architecture may represent the higher-level interactions between clusters within a group, and a non-fully connected architecture may represent interactions between groups of clusters.

These exemplary fusion algorithms allowed system performance to be evaluated for alternative sensing and target detection algorithms, control algorithms, survivability strategies, and agent paradigms in the face of a wide variety of operational challenges presented to the distributed system [32]-[36]. The EIF algorithms were applied to the tracking problem represented by (7) and (8) for sensors that report range and bearing. However, the algorithms could also be applied to a tracking problem representation developed for measurements that are the (x, y) target position estimates computed as described in the *Maximum Likelihood Target Localization Using Binary Sensor Data* section of this report.

Fuzzy-Reinforcement Learning to Track a Mobile Target using a WSN

Fuzzy logic and reinforcement learning techniques were applied to the problem of predicting and tracking the position of a mobile target as it travels through a distributed WSN [37]. The accuracy of the target position prediction, amount of communication between distributed sensors, and power consumption are all issues that influence the performance, reliability, and survivability of the sensor field. A reinforcement learning method based on fuzzy logic was designed to improve the accuracy of an existing target tracking method [38] without adversely affecting its underlying communication and power consumption. The reinforcement learning method incorporates a feedback error term that represents the estimated difference between the actual and projected target positions. This error term continuously adjusts a fuzzy inference mechanism that has been added to a previously defined tracking approach [37]. The new approach was compared to the unaugmented tracking approach for a variety of target movement

scenarios in simulation. Simulation was used to compare Fuzzy reinforcement learning improved the mean square tracking error (MSE) except in cases where the target maintained a nearly constant velocity for which fuzzy reinforcement learning and the unaugmented approach both produced a low MSE.

The fuzzy-reinforcement learning method is an example of an intelligent systems approach to target tracking. Extending this fuzzy-reinforcement learning method to counter the effects of sensor noise and sensor placement inaccuracies and the application of Neuro-Fuzzy techniques to train and optimize the initial fuzzy sets are areas for future research.

Control and Coordination for Optimizing Performance and Life

As described in a previous section of this report, the strategies used to increase survivability in this work include 1) incorporating and utilizing redundancy 2) balancing resource utilization among available reserves, and 3) using local optimization to improve global performance. This section describes several approaches in which these strategies are brought to bear in optimizing performance and life. Redundancy may be available in the form of underutilized existing network resources or it may be provided by the placement of extra resources that would not otherwise be available in the network. These approaches utilize redundancy to support the resistance, recognition, and recovery properties of the sensor network to increase survivability and to increase overall performance and life of the field.

Redundant Functionality

If redundant functionality is incorporated in the network, then reconfiguration could make use of existing resources by moving the function of a dead or dying node to another node having the capability to execute the function. For example, if the power reserve falls below a threshold in the node currently handing master node functions such as data fusion and communications with the outside world, those functions would be relocated to another capable node.

In one approach, the capacity to perform the master level functions in the computational hierarchy was distributed among several cluster nodes and the coordination function was used to move the master level functions from one capable cluster node to another to increase the lifetime of the field [33], [35], [36]. The cluster nodes ran cluster level functions unless they were also called on to run master level functions. Master level functions running on a cluster node utilize additional computational and communications resources, and need more power than is required for the cluster level functions alone. The routing of communications to and from the cluster node running the master level functions increases power utilization at surrounding sensor nodes. If the master level functions would always be carried out at a single cluster node, the resulting power drain would most likely cause this cluster node or one of the nearby sensor nodes to die first due to battery depletion, ending the useful life of the field at a time when most other nodes would be likely to have significant power remaining. With the approach of moving master level functions, the resource utilization was balanced among the cluster nodes so that the field degraded gracefully and the field lifetime increased.

Balancing the Utilization of Communications Resources

An approach for balancing the utilization of communications resources throughout the network by routing based on power and node availability preserved the availability of the resources needed to route messages throughout the network. Several schemes were considered for routing based on power and node availability [39]. Preservation of critical communications resources would increase the useful lifetime of the field.

Communication requirements both internal and external to the cluster were considered. In this work, it was assumed that sensors could communicate directly with their cluster nodes. Communication between clusters was needed in varying degrees to support fusion, tracking, classification, queuing, switching modes of operation, and information exchange needed to maintain redundancy. Communications between clusters may also include environmental information, information about likely targets in the area, as well as additional information external to the field that can be exploited to control processing at the cluster nodes and provide the best opportunities for correlation of sensor detections. Since cluster and master nodes were spread over a wide area, cluster-to-cluster relaying of information was used to transmit information between master nodes, cluster nodes and the external command center. The node sending the data communicated with the receiving node through cluster-to-cluster relaying of information using intermediate sensor nodes. Each node maintained a routing table that contained information for routing data to a given receiver node.

In the Fixed Multi-Hop Routing (FMR) protocol, the data from the sending node is routed to the receiving node through intermediate sensor nodes. Each node maintains a routing table that contains information for routing data to a given receiver node. The Probabilistic Multi-Hop Routing (PMR) protocol attempts to equalize the routing load uniformly among all the nodes in the sensor network. In contrast to FMR, where every node has only one next-hop node for routing data, in PMR one of several possible next-hop nodes is selected randomly by the sending node at run time. The Modified Probabilistic Multi-Hop Routing (MPMR) protocol is similar to PMR, except that while calculating the next-hop node from the pool of possible next-hop nodes, the previous use of nodes (state) is taken into account. In other words, a node's probability of selection is reduced if the node was used previously for routing data using MPMR.

For each of the FMR, PMR, and MPMR protocols discussed above, whenever there is a need to update the routing table due to factors such as low battery energy at the routing nodes, the resulting communication overhead is aggravated by redundant communications between nodes. Localized Optimization (LO) is a new scheme developed in this work that attempts to reduce this overhead. LO exploits the fact that a cluster node a) has access to the information about battery energy levels at the sensor nodes within the cluster with negligible communication overhead and b) possesses the required computational power to perform a localized routing adjustment. In LO, whenever a routing node's battery energy level falls below a preset threshold, the cluster node employs a set of rules to choose another node in the cluster to act as a replacement routing node. This scheme eliminates the need to inform other clusters in the sensor network of the new routing node, resulting in a substantial decrease in the communications overhead from that required to update routing tables throughout the network. Since the LO concept involves only changing the network address and range setting of a given sensor node, the

scheme requires very little computational and communication overhead. The LO concept was applied to each of the FMR, PMR, and MPMR routing protocols.

The routing protocols were implemented and tested in simulations using resource-based modeling. The field lifetime was the time from the start of the simulation until battery energy power was exhausted at any node. The overall field lifetime was higher with PMR and MPMR than for FMR. However, since PMR and MPMR select the next routing node randomly, the next routing node selected may sometimes be the same node that forwarded the data to the current sending node, resulting in network oscillations. Due to these network oscillations, the time taken to route data in PMR and MPMR was higher than FMR, resulting in delays in data transmission that may adversely affect performance in some applications such as target tracking. The use of LO improved the field life for all of the routing protocols. The improvement in the field life was due to the effective use of the inherent redundant resources available in the sensor network. However, LO also introduced delays in data transmission. The use of FMR with LO is the most promising of the routing protocols considered, since it significantly improved field life at the expense of only a minor increase in delays, and can be applied locally within a cluster [39].

Balancing the Utilization of Power Resources

The cycling of the power supplied to some sensors based on the need to acquire measurements for tracking provided a simple means to conserve power. The evolution of the covariance of the track estimate computed using the EIF algorithm served as a useful means to determine when to turn on a sensor in order to acquire the measurements needed to maintain a given tracking accuracy [36].

Pre-Computed Reconfiguration

Evolutionary programming, ant algorithms, and immune algorithms are examples of emerging biologically inspired approaches that may find application in optimization, distributed control and eliciting cooperative behaviors. Some of these methods are now being applied for combinatorial optimization and routing in communications networks. We investigated evolutionary programming and ant algorithms for message routing using the traveling salesman problem [40]. Ant algorithms generally outperformed genetic algorithms (GA) in computing optimal routes, but required more computation.

However, optimal reconfiguration in real time during normal system operation is difficult and requires significant communications and computations. Therefore, it may not be practical to run the optimization algorithms needed to perform complex reconfiguration in real time, or at the time when the network needs reconfiguration. When a scenario requiring reconfiguration is recognized, the use of a pre-computed reconfiguration solution that fits the recognized scenario could allow reconfiguration to take place without requiring the real time execution of computationally intensive optimization algorithms.

We investigated methods for pre-computed reconfiguration that consider the layout of the deployed sensor field in computing a set of configuration and operating options that are optimal for given scenario permutations (combinations of factors including loss of clusters, low power reserve at a cluster, externally supplied operating mode, and disturbed communications) [40]. A

Genetic Fuzzy Rule Base System (GFRBS) was used to create and optimally tune fuzzy rules from an initial population of rules. In the five-node problem studied, the resulting rules yielded the optimal route if one node was lost. If the autonomous entity (node, agent) is able to recognize a new scenario that requires reconfiguration, the system may be able to apply the pre-computed rules determined by the offline tuning of the GFRBS to reconfigure itself to an optimal configuration that is appropriate for the recognized scenario. Potential benefits of pre-computed optimal configuration include a longer field life due to reduced power consumption, a reduction in the likelihood of detection by the enemy due to reduced communications during reconfiguration, a wider array of reconfiguration options than is possible if optimization must be computed in real time, the use of more powerful optimization algorithms, and a more seamless transition to a new configuration.

Performance Guided Reconfiguration

The initial network design and layout of a sensor field considers the required network performance [41]. However, as sensors fail due to battery exhaustion or other reasons, performance degrades and detection, localization, or tracking coverage holes will develop unless additional sensor resources maintain the required performance. On the other hand, field performance requirements may not be constant over all phases of field lifetime. If performance requirements decrease over some period, unessential sensors could be turned off to conserve resources. If performance requirements increase, the new performance goal could be met by awakening redundant sensors as needed. We developed performance-guided reconfiguration (PGR) to reconfigure the field to enable performance requirements to be met while conserving resources.

This work considered the application of PGR in a sensor network designed to localize a target based on binary detection reports from sensors arranged in clusters. A PGR algorithm was applied to reconfigure a sensor network by awakening dormant redundant sensors as needed to meet desired performance goals when sensor failures occur [42]-[44]. Redundant sensor nodes placed in the network were dormant until they replaced a failed sensor. When a sensor fails, PGR identifies candidates for replacing the failed sensor from the set of available redundant sensors, and uses a performance-based cost function to select the candidates to be activated [42]. The PGR algorithm is applied locally within the cluster that has a failed sensor. Some initial simulation results of localized PGR, in which PGR is applied at the cluster level in a multicluster sensor network, were described in [44].

In PGR, multiple sets of redundant inactive sensors that could be activated to meet the performance criterion within the cluster are initially identified as candidates for replacing the failed sensor. The candidate sensor set that optimizes cluster performance is selected using a cost function that weighs additional criteria of interest. The cost function provides the basis for ranking all possible solutions and for adjusting the weights of the individual performance objectives in terms of the total cost for a candidate set S.

$$Cost(S) = N_s c + \sum \gamma(r_i - r) + \eta \sum_{j=1}^{4} \frac{N_j}{R_j} \tag{9}$$

In (9), γ and η are adjustable constants, N_s is the total number of sensors in candidate set S, c is a flat cost per sensor, and r is the radius of a no-penalty search area around the dead sensor. The distance of candidate sensor i from the dead sensor is r_i, N_j is the total number of sensors from candidate set S that are in area j, and R_j is the number of redundant sensors available in area j (area refers to one of four sectors formed by dividing the area around the dead sensor bounded by the search radius r into four parts).

The three terms appearing from left to right in the cost function are referred to as term one, term two, and term three. The cost function weighs the **absolute sensor cost** (term one), the **distance of the candidate sensor from the failed sensor** (term two), and the **cost of utilized redundant resources** (term three). Term two is applied only if the candidate sensor is outside a no-penalty search area such that r_i-r is positive. A no-penalty search area is an area bounded by the initial search radius. The search radius is increased if sensors in the area bound by the initial search radius do not meet the performance requirement, and term two is then non-zero.

For a simple demonstration of localized PGR, we compared the performance of a multicluster network using localized PGR, a nearest-neighbor reconfiguration approach, and an approach without field reconfiguration [44]. Without PGR, coverage performance degraded as sensors were lost. With PGR, every time a sensor failed, the respective cluster was reconfigured to maintain coverage. In this way, clusters made local reconfiguration decisions to achieve the global field performance goal. A more detailed example of PGR is described in the **SIMULATION** section of this report.

Determining whether PGR is needed after loss of a sensor requires the intensive computation of the CRLB at many potential target locations within the cluster. Also, the process of determining the candidate sets of sensors that will attain the required performance is computationally intensive. Developing more efficient methods for computing the CRLB and for carrying out PGR using pre-computed reconfiguration are topics for future work. Other topics for future PGR work include the development and simulation of algorithms for turning sensors on and off as performance requirements change over time, and optimizing the placement of redundant sensors, especially in the area where clusters overlap.

V. INTELLIGENT SOFTWARE AGENTS

An agent is an entity that acts in the place of another in order to bring about a desired result. Agent systems and multi-agent systems are a new and robust paradigm for designing flexible architectures for systems with disparate needs. Agents are especially suited for environments that are open and distributed [16]. An agent is characterized by the concepts of situatedness, autonomy, and flexibility. Multi-agent systems consist of several interacting agents and are appropriate for cases where data are distributed and incomplete, individual agents have a limited viewpoint, there is no global control, and computation is asynchronous. Multi-agent systems provide a level of abstraction that enhances computational efficiency, reliability, extensibility, maintainability, responsiveness, flexibility, and reuse. With these properties, multi-agent system technology appears to be an ideal candidate to realize autonomous distributed system designs and provides a framework that may be useful in achieving survivability [15], [45].

In this work, agents provided intelligence within the system with a goal of increasing the network field life without adversely affecting performance [34]-[36]. A multi-agent system was realized by wrapping selected cluster nodes and master nodes with an agent, with all communication and system actions routed through the agent. Agents supported actions within nodes and interactions between nodes in many ways. Agents facilitated the distribution of information, facilitated the distribution of the data fusion, control, and coordination functions, supported achieving survivability, and provided a high degree of autonomy when needed.

A scheme of 'functional agents' was used to allow agents to carry out multiple functions at a node [35]. Functional agents performing different functions were stacked on the same physical hardware. As a result, there was an increase in the number of nodes in the system that could perform the various functions and the hardware requirements were simplified. Thus, stacking functional agents is a method that uses the abstraction provided by agents as an alternative for simplifying hardware requirements in sensor networks.

A variety of agent paradigms applicable for resource management was developed and compared using simulations [34]-[36] (see appendix). Such a comparison provides an increased understanding of how the performance of the sensor network changes with the number, functionality, and presence of agents at different levels in the network hierarchy. A modular simulation framework based on object-oriented design was developed to facilitate the implementation and comparison of different agent-based scenarios. This framework was used to generate data for detailed analysis of component interactions and for evaluation of the integrated system. Results compared the number of computations, number of communications, tracking performance, and field life for Monte Carlo simulations of scenarios in which different agent paradigms were employed. These comparisons should assist designers of agent-based systems in utilizing agents to their best advantages in scenarios similar to the wide range of those explored.

We also developed a measure of the machine intelligence of an agent-based system, the Machine Intelligence Quotient (MIQ) [34], [36]. The MIQ may make it possible to predict the performance of agent-based alternatives having different complexity. Such predictions may be useful in deciding from among the myriad of different agent-based alternatives. In its present form, the MIQ is quantitative. Refinements to introduce indicators that give a better reflection of quality are topics for future work.

Another area for future work is the use of cooperating agents in supporting survivability through enhancing the network's ability to adapt and evolve to reduce the effectiveness of future threats. Agents might recognize the type of attack or accident that caused damage and take steps to either reconfigure the network or adjust operating modes in order to reduce future vulnerability to that threat.

Gliders as Agents for Undersea Data Collection

In another research thrust, gliders were considered for data collection in undersea WSN applications [46]. Undersea data collection is a significant research challenge since it must accommodate the power and bandwidth constraints of the WSN infrastructure and the specific needs posed by the application. Autonomously navigating gliders provide a possible solution for data collection tasks in civilian and military applications. The most important factors in the use of gliders are navigational speeds, routing, the structure of the WSN hierarchy, and the effect of oceanic currents. Glider characteristics important for use in data acquisition for target detection were reviewed. Some exemplary glider routing algorithms were developed and tested in simulations. Simulations were used to compare the time required for the glider to collect data from all sensors in each routing scheme, consider the effect of oceanic currents, and illustrate the behavior of gliders under different network topologies and assumptions. The speed at which current gliders travel introduces delays in acquiring data from the sensor with longer delays occurring for more distant sensors.

The literature review raised issues such as the effect of oceanic currents on performance. It was observed that, irrespective of the direction of glider movement in reference to that of the currents, the glider must reduce its speed to maintain its course and navigational accuracy.

Routing a glider in single-tier networks requires the glider to visit each sensor in the network. Hierarchical sensor networks use a cluster paradigm in which the sensor nodes report their data to their respective cluster nodes. The glider can be made to visit a cluster node in such hierarchical networks to provide several advantages. Visiting only one cluster node instead of each sensor node saves the glider significant energy and time in collecting data from the entire network. The effect of oceanic currents, obstacles, or internal positional error might change the course of the glider to affect its ability to visit all the sensor nodes on the intended path. In hierarchical network structures, the glider visits the cluster node located at the center of each cluster. If the glider loses its bearing, then it might reach any sensor nodes on the periphery of the cluster and receive the information updates for that entire cluster. Simulations of glider routing in a two-tier hierarchical structure provided more flexibility, reliability, and speed of data collection from the sensor nodes than can be achieved in single tier networks.

This work is a step towards developing realistic scenarios and assessing glider performance under different circumstances that will enhance the application of glider technology for data acquisition in sensor networks. Future work may include adaptive routing in which the glider dynamically changes its route as it processes the sensor data. Various assumptions made in developing the simulations could be eliminated in the future when more accurate data becomes available e.g. the deployment of the sensors can be assumed to be random instead of symmetric. The research could be extended to explore and simulate additional applications such as hardware reconfiguration of sensor networks in which the glider can physically replenish dead sensors.

VI. SIMULATION

Simulation was used to develop and test alternative Micro-DADS realizations and algorithms, consider tradeoffs, and evaluate the ability of the distributed system to achieve its goals when presented with a set of operational challenges. For initial evaluation of each Micro-DADS alternative, high level and simplified representations of fusion, classification, control, communications, and other processes were employed. As new alternative algorithms, strategies, and agents were developed, they were integrated and tested in simulations. The utilization of battery energy, communication, and computational resources was modeled at each system node in order to provide the opportunity to study tradeoffs involved in designing power efficient coordination, communication, and reconfiguration strategies.

Underwater Target Tracking Using Sensors that Report Range and Bearing

Three-Tier Hierarchical Distributed Sensor Network

Initially, a modular system level simulation of a three-tier hierarchical distributed sensor network was developed for the underwater target tracking application [33]. The three-tier network shown in Fig. 7 includes sensor nodes (small circles) that report measured range and bearing of submarine targets to cluster nodes (black dots) that perform local data fusion. The large circles represent cluster nodes at which the master function may be carried out. At a given time, one of the cluster nodes carries out the master function (diamond) by gathering the data from the cluster nodes and performing global data fusion, tracking, and communication with the outside command center.

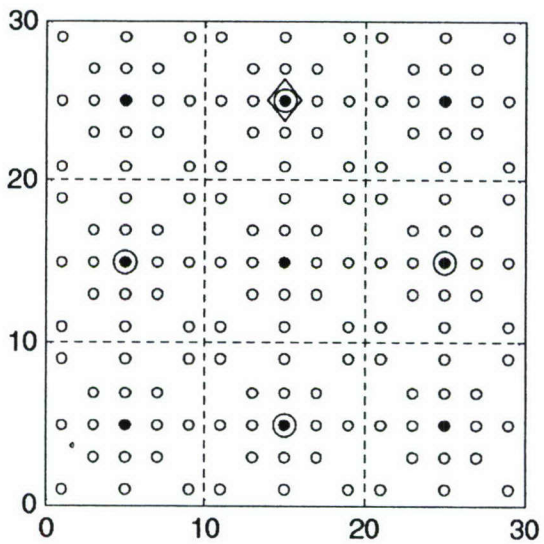


Fig. 7. Three-tier hierarchical sensor network architecture.

The Node Model

A node model was used to simulate events and resource consumption occurring at each node (Fig. 8). In the system level simulation, a field layout is initially specified, the target track is generated, and events are simulated at each time step. First, the target position is related to the range of each sensor to facilitate measurement of the target position by the sensors. Next, a communications model runs to transfer information from one node to the other. Next, the node models are run.

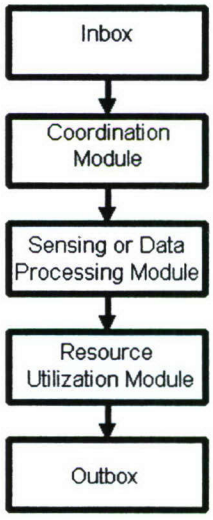


Fig. 8. Operational flowchart of a node model.

The initial Micro-DADS realization utilized a three-level computational structure that includes an organization level, a task planning level, and a task execution level at each node. A fuzzy rule-based inference system and other advanced methods were considered for use in task planning, but offered no benefits for the level of implementation detail used in this research. At the beginning of each time step, a node's organization level receives information such as commands, communication counts, and mode of operation from the communications module through an inbox. Next, at the node's task planning level, a coordination module is run to coordinate events based on the received information. In a sensor node, the coordination module controls simple functions such as the sleep-awake mode of the sensors. In a cluster or a master node, the coordination module completes tasks such as synchronizing, organizing, or reconfiguring the network. Next, at the execution level, a sensor node executes the sensing module, and a cluster or master node executes data-processing module to run data fusion algorithms based on distributed EIF filtering [27], [33]-[36].

After execution of the sensor or data processing module that is appropriate for a node, the resource utilization module executes in order to account for the power used by the communications and computations occurring during that time step. The following section presents more detail about the resource utilization module.

The communications model is responsible for representing salient features of the sensor network communication. A simple outbox-inbox communication scheme transfers the data

between the nodes. The data is transferred from the transmitting node outbox to the destination node inbox. At the end of a time step, the node sends information to its outbox for transmission. The communications model takes the information from outboxes and sends the messages using a communications protocol and appropriate routing to the appropriate inboxes. Hence, at the fastest, messages placed in an outbox at simulation instant t will be available in the destination inbox at simulation instant $t+1$. However, computations at the routing nodes, network bandwidth, environmental problems, multihop routing, and queuing at the sensors in the routing path may introduce delays in the communication of information to the destination. Delays may affect various aspects of the system performance. For example, communication delays would degrade the target tracking performance. The communications model introduces communications delays due to the above factors.

Resource-Based Modeling

Network performance and lifetime depend on the battery energy, communication, and computational resources available at each system node. The first step in optimizing performance and lifetime involves analyzing the sensing, data processing, communication, and coordination modules of a node and modeling their impact on utilization of battery energy, communication, and computational resources. We developed a resource-based modeling framework for system level simulation of sensor networks to offer flexibility and fast generation of various operational network scenarios. This simulation framework provided the opportunity to study tradeoffs involved in designing coordination, communication, and reconfiguration strategies that preserve power, communication, and computational resources [33].

The computational and power resources of each node utilized by sensing, data processing, communications, and coordination events are tabulated. Resources utilized by communications are tabulated for each node by the communications model, and those utilized by sensing, fusion, and coordination are tabulated at each node using the resource utilization module (Fig. 8). Proportionate battery draining weights are assigned to these tabulated events depending upon the actual practical node model used for the simulation. Based on the assigned weights, the power dissipated and the remaining battery energy are computed for each node at each time step.

Power utilization drains the battery and affects the node and field life. Different battery models were evaluated in order to allow power utilization to be computed accurately. A stochastic model of the battery capturing the essence of the charge recovery mechanism was implemented to account for the relaxation phenomenon and the rate capacity effect [33].

Coordination schemes were developed in this work with the aim of prolonging the sensor network lifetime by efficiently utilizing available resources. These schemes allowed reconfiguration of the sensor network to make more optimal use of available resources and to take advantage of redundant resources. Simulation results show that reconfiguration of sensor networks improved performance and lifetime over that of a sensor network having a fixed configuration.

Object-Oriented Agent Simulation

The initial system level simulation of an underwater distributed sensor network was used as

the basis for developing an object-oriented framework for simulating agents in a distributed sensor network [34]-[36]. The framework is not restricted for use in this application, but can be easily adapted for simulating agent-based sensor networks for other applications. Since the system under consideration is large, having a large number of entities, different implementation strategies, and detailed interactions between the different entities, simulations were developed using a modular approach. Software modules were developed to mimic the entities and processes that would be present in an actual physical system. The system was partitioned into components that can be modeled at the lowest behavioral levels. This modular approach gives the flexibility and extensibility of changing, adding and removing modules for simulating different scenarios and strategies at a fast pace. This framework utilized the advantages of object-oriented programming techniques in system design. This framework allows easy scaling of the sensor network. Nodes can be added or removed, different communication and battery models can be simulated, and different algorithms can be easily employed within the agents thus facilitating the simulation of a large number of scenarios in a short time.

Fig. 9 shows the object diagram of the system that includes multiple master, cluster, and sensor nodes. Agents are present on the master and the cluster nodes. The sensor nodes report measured range and bearing of target to cluster nodes that perform local data fusion. For a system that does not include master nodes, some cluster nodes will include a master agent in addition to the cluster agent.

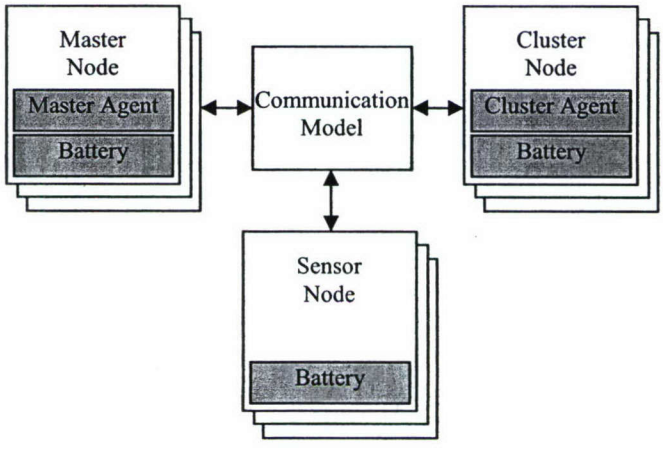


Fig. 9. Object diagram of the simulation framework.

For the outbox-inbox communication, a localized optimization scheme was used to route the data through the network [39]. Localized optimization uses the fact that cluster nodes have access to information regarding the residual power levels of the cluster's sensor nodes. This information is used within the cluster for localized routing reconfiguration.

Operational challenges included commands specifying various operating modes and theater information received from the command center, a rich set of target tracking and classification scenarios, undersea environment, field configuration, field topology, loss of clusters and sensors, loss of communications, computational load, and data transmission latency. Simulation was used

to provide insight into the parameters that impact overall performance of Micro-DADS, expose trade-offs, allow design alternatives to be explored, and facilitate the identification of requirements. In order to develop a quantitative evaluation of the performance of decentralized alternatives, decisions were compared to those of a hypothetical entity that makes decisions using a centralized paradigm based on the exact and current knowledge of every state of every entity in the entire system [7].

Visualization

We explored the use of visualization and virtual reality environments to provide the visual effects of real scenarios as a tool to understand, interpret, evaluate, and compare tradeoffs and efficiency. The Enabling Technology Laboratory (ETL) in the UAB Mechanical Engineering Department collaborated in the visualization work. The ETL has a 128 processor LINUX cluster and visualization infrastructure. This includes a 9-tile Viz-Wall with 9 associated processors projecting a very large-scale high resolution image in a synchronized fashion, and a Viz-Box with high performance processors to perform virtual reality in a three dimensional environment. A high-end server computer acquired for this project was employed as a database server on which visualization queries were performed. Custom visualization algorithms tailored for Micro-DADS interrogation were developed and validated for simulated data.

We used bathymetric data to develop a visualization of target tracking in an undersea environment [47]. A three-tier network using sensors that measure range and bearing was used (Fig. 7). In order to visualize implementations of the tracking algorithms developed using MATLAB, pertinent parameters and output data were extracted from the sequential functional simulation and stored in a text data file. The data file lists the positions of the sensors, pattern of the clusters, coordinates of the target and corresponding estimated positions at each time step. This data was loaded into the program developed to visualize the seabed and then the simulation was performed using frames to denote the target position at each time step. The mouth of the Chesapeake Bay was used as the seabed for the initial development. Glyphs were used to denote various items in the simulation, including sensors, the range of detection of sensors, a submarine target, the history of the target position, the current estimate of target position, and the history of the estimate (Fig. 10).

Using the FLTK graphical user interface toolkit, a graphical user interface for this visualization was built to enable a nice interaction with the algorithms. In order to offer a high-resolution visualization and to have stereoscopic display the developed OpenGL-based visualization code was ported into Vis-Wall and Vis-Box facilities available at the ETL. Future work could utilize the Vis-Box for a fly-through view of the total undersea region of interest. The footprint of the Vis-Box is 8x8 feet and it is a few inches shy of being 8 feet tall, making it close to an 8x8x8 foot cube. With this system, researchers could visualize their data in the stereoscopic virtual environment. Imagery is projected to the rear of a semi-rigid or flexible screen, which is mounted to the front of the unit. Applications software would be used to generate separate images for each eye. Users wear lightweight, inexpensive polarized eyeglasses and see a stereoscopic image.

Fig. 10. Visualization of the tracking of a submarine in the mouth of Chesapeake Bay using ten clusters with each cluster having two sensors. The actual target path is shown in gray with the sensor based target track estimate in white. The cluster having the target currently within its range is highlighted in dark gray.

Target Localization Using Sensors that Report Binary Detections

The object-oriented simulation framework for distributed sensor networks was extended to use sensors that report binary detections and to use the ML estimation framework for target localization as described in the earlier *Maximum Likelihood Target Localization Using Binary Sensor Data* section of this report [19], [25]. A detailed simulation was performed to demonstrate Performance Guided Reconfiguration [42]-[44].

Simulation of Performance Guided Reconfiguration

The use of performance-guided reconfiguration (PGR) was studied in simulation. When sensor failures occurred, PGR was used to awaken dormant redundant sensors as needed to meet performance goals. The sensor field layout corresponds to a 9-cluster distributed sensor network consisting of 49 binary sensors in each cluster (Figs. 11-12). Cluster and master nodes are not shown. The field includes 441 active sensors at integer positions and 324 redundant sensors between active sensors. Clusters were positioned to overlap with neighboring clusters such that the nominal sensor field would satisfy a performance criterion for all target locations within the coverage area shown bounded by the dotted line. The gray areas show overlapping sensor positions between neighboring clusters. The area shown by the dotted squares denotes the coverage area, a square of side 12 units.

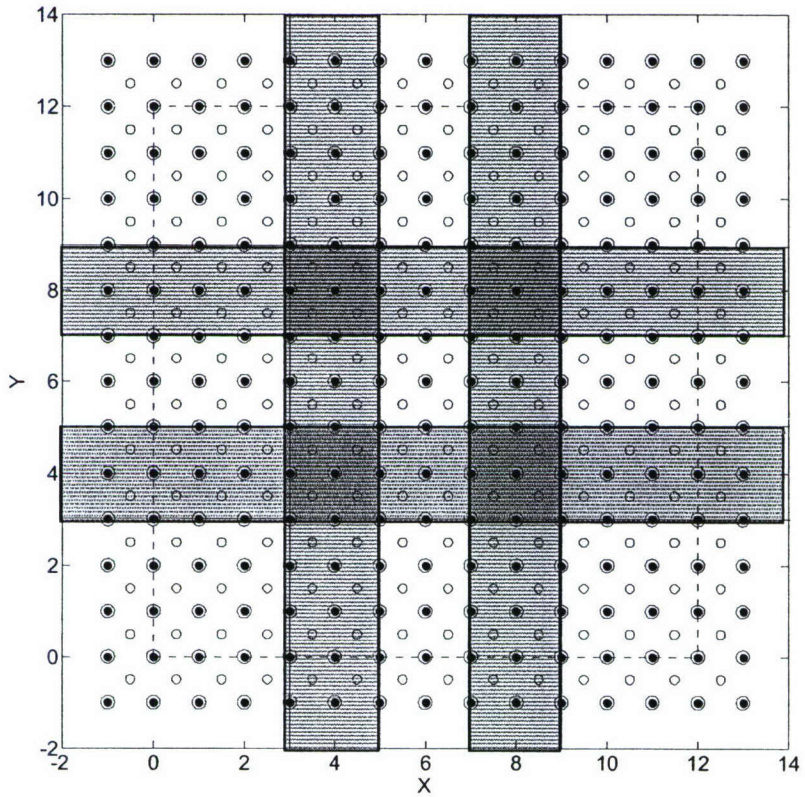


Fig. 11. Layout of the 9-cluster field with 441 active sensors at integer positions shown as double circles, and 324 redundant sensors between active sensors shown as smaller circles. The area shown by the dotted square denotes the coverage area, a square of side 12 units. The gray area shows overlapping sensor positions between neighboring clusters.

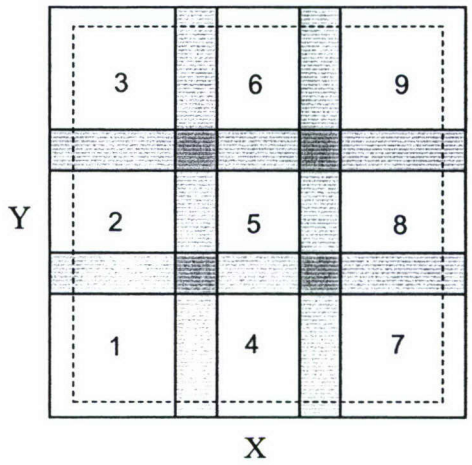


Fig. 12. Layout of the field showing overlapping numbered clusters. The area of overlap, shown in gray, is 2 units wide. The area inside the dotted line is the coverage area.

The target was made to appear randomly throughout the coverage area of the field, and assumed a new position every 25 simulation instants. At every simulation instant, the sensors made binary decisions about the presence of the target for each of ten different thresholds. The target localization was done using the two-step approach [25]. Based on binary sensor decisions from all ten timeframes, the respective cluster nodes first formed a weighted average (WA) estimate of the target position. This WA estimate of the target position was then used as a starting point for a Nelder-Mead search to obtain the final target position. This two-step approach for target localization helped to achieve a localization performance at the CRLB.

The power consumption of all the nodes in the field (sensor, cluster, and master) was considered using resource-based modeling. The assigned weight for power utilization was highest for communications transmission, followed by the weights for sensing, receiving communications, and maintaining an awake state. As the simulation progressed, node batteries were depleted depending on the respective activities. In order to speed up the execution of the simulations for this work, all sensor node batteries were initialized at 10 percent of full energy capacity at the start of the simulation. The cluster and master node battery capacities were initialized to high values to guarantee that they would not die prematurely to end the field life, insuring the simulations provide a good demonstration of improvements offered by PGR.

Sensor decisions were routed to respective clusters using one-hop communication. Clusters used multi-hop communication to communicate decisions to the master node. The routing algorithm was PMR. Each node in the sensor network (sensors and clusters) maintained a routing table formed at the beginning of the simulation. Routing tables were updated every time that a sensor failed [39].

When a sensor failed, a decision of whether or not PGR would be required to run was made. The CRLB for the RMS localization error in target position estimate, $\hat{D}$, was computed at 1681 positions (41x41) within the cluster that included the failed sensor. The CRLB for each target position evaluated was then compared to an RMS error limit of 0.1610. This limit served as the maximum allowable value for the computed RMS error. If the CRLB for any of the evaluated

positions exceeded the limit, it was determined that reconfiguration was necessary, and only then, localized PGR was run in the respective cluster.

Localized PGR resulted in the choice of the optimal redundant sensor set that would be awakened to replace the dead sensor. PGR evaluated all the possible sets of replacement sensors available in the vicinity of the dead sensor, and computed the cost of every set that could meet the limit of 0.1610. The cluster was then reconfigured by waking up the sensors belonging to the set that had the lowest associated cost. A representative cost function was utilized. The weights for the constants in (9) were chosen to be equal, with that for resource utilization ($\eta = 0.1$) equal to that for performance ($\gamma = 0.1$).

In order to show the field performance at the start of the simulation, the initial CRLB surface was computed at 1681 positions (41x41) within each cluster to show the achievable localization error limit for various target positions using the nominal layout of active sensors (Fig. 13a). The performance criterion was met for all target locations considered (Fig. 13b).

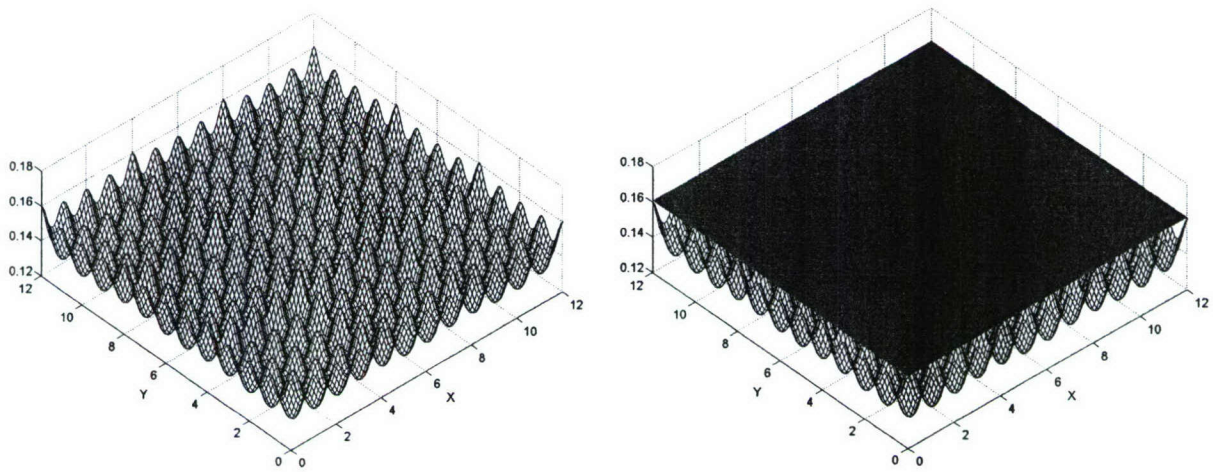
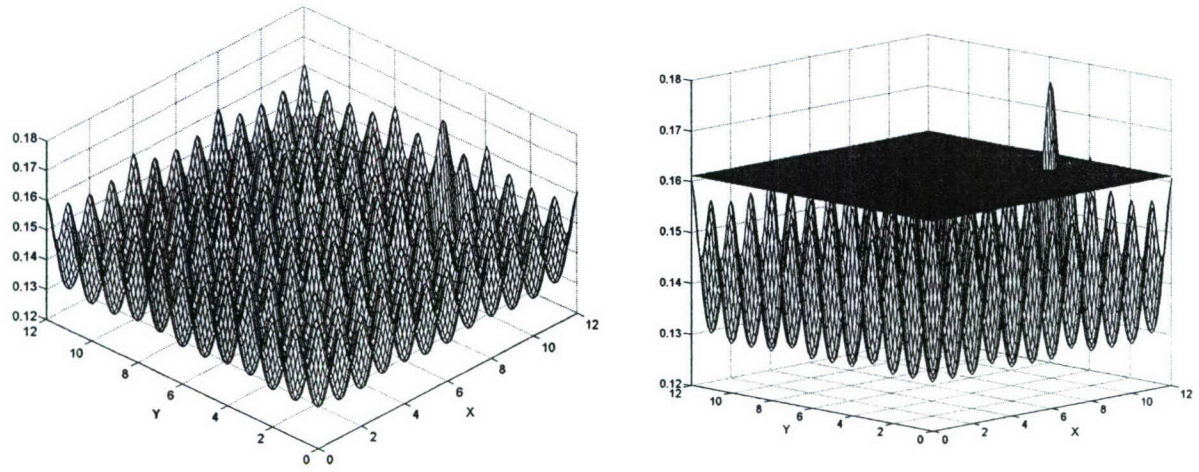


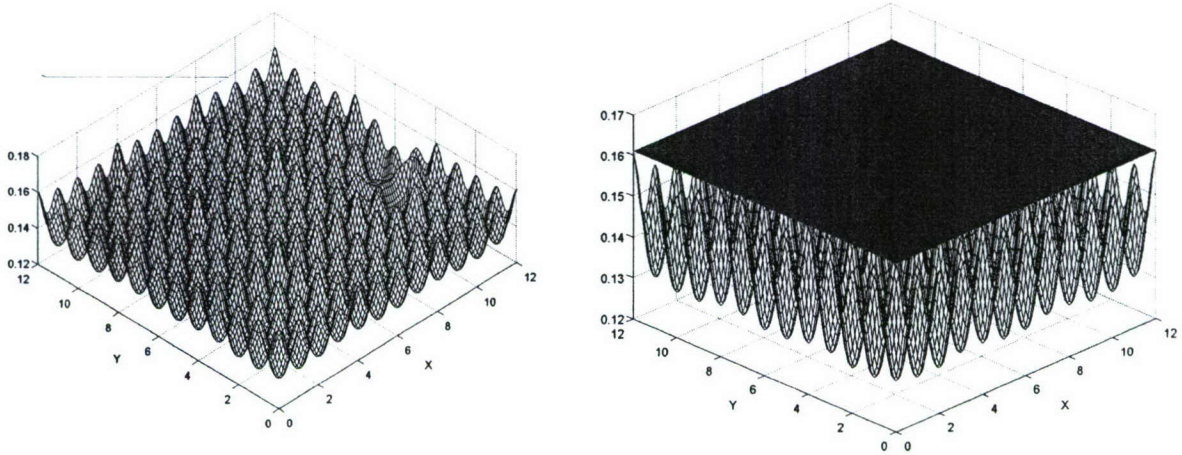
Fig. 13. (a) Computed CRLB for the RMS error in the estimate of D versus X and Y for the field with 441 active sensors at the beginning of the simulation. (b) Computed CRLB with the performance criterion of 0.1610 for the RMS error shown as a plane. The entire coverage area met the performance criterion, which is indicated by the fact that the CRLB surface is bounded by the plane for all target positions inside the coverage area.

The CRLB surface was computed at several representative simulation instants to show the effects of failed sensors and to show that performance was restored with PGR. The first sensor was lost at simulation instant 1473 from cluster 7, with a consequent loss in performance such that a fraction of the coverage area near the failed sensor did not meet the performance criterion (Fig. 14). PGR reconfigured the network by activating redundant sensors in order to allow the entire coverage area to meet the performance criterion (Fig. 15). As additional sensors were lost during the simulation run, the field performance was compared to the criterion, and PGR was used to replace lost sensors if required to allow the performance criterion to be met.



(a) (b)

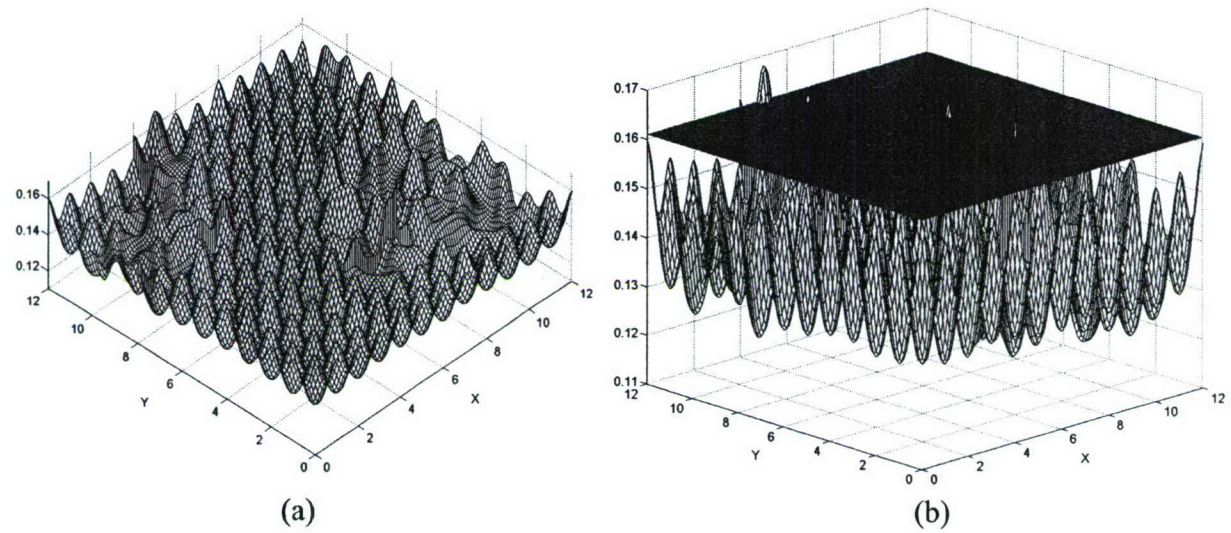
Fig. 14. (a) Computed CRLB for the RMS error in the estimate of D versus X and Y for the field after the death of the first sensor in cluster 7 at simulation instant 1473. (b) Computed CRLB with the performance criterion of 0.1610 for the RMS error shown as a plane. A fraction of the coverage area near the failed sensor did not meet the performance criterion (shown by peaks above the surface of the plane).



(a) (b)

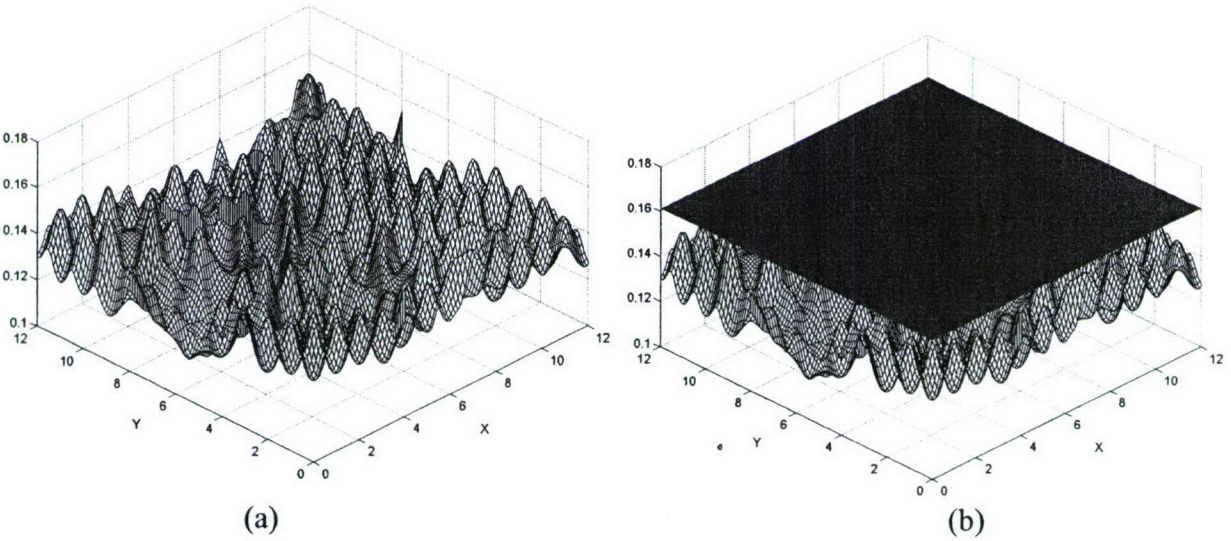
Fig. 15. (a) Computed CRLB for the RMS error in the estimate of D versus X and Y for the field after the death of the first sensor in cluster 7 at simulation instant 1473, but with nearby redundant sensors awakened using localized PGR. (b) Computed CRLB after PGR with the performance criterion of 0.1610 for the RMS error shown as a plane. The entire coverage area met the performance criterion, which is indicated by the fact that the CRLB surface is bounded by the plane for all target positions inside the coverage area.

Five sensors died in clusters 5, 6, and 8 at simulation instant 1645, with a consequent loss in performance such that a fraction of the coverage area near the failed sensor did not meet the performance criterion (Fig. 16). PGR reconfigured the network to allow the entire coverage area to meet the performance criterion (Fig. 17).



(a) (b)

Fig. 16. (a) Computed CRLB for the RMS error in the estimate of D versus X and Y for the field after the death of five sensors in clusters 5, 6 and 8 at simulation instant 1645. (b) Computed CRLB with the performance criterion of 0.1610 for the RMS error shown as a plane. A fraction of the coverage area near the failed sensors did not meet the performance criterion (shown by peaks above the surface of the plane).



(a) (b)

Fig. 17. (a) Computed CRLB for the RMS error in the estimate of D versus X and Y for the field after the death of five sensors in clusters 5, 6 and 8 at simulation instant 1645, and nearby redundant sensors awakened using localized PGR. (b) Computed CRLB after PGR with the performance criterion of 0.1610 for the RMS error shown as a plane. The entire coverage area met the performance criterion, which is indicated by the fact that the CRLB surface is bounded by the plane for all target positions inside the coverage area.

The simulation was terminated at simulation instant 1841 because the redundant resources in one of the clusters (cluster 8) were exhausted by PGR. However, the entire coverage area met the performance criterion at instant 1841 (Fig. 18).

Thus, the results show that PGR used available redundant resources to increase the time over which the performance criterion could be achieved throughout the field. The time over which PGR could be used to allow the performance criterion to be met over the field would increase if additional redundant sensors were placed in the field. If the simulation would have been continued past simulation instant 1841, PGR could have been applied in clusters where redundant resources remained, but PGR could not have been applied where no redundant resources remained. Eventually, as additional sensors were lost in clusters in which PGR could no longer be applied because redundant resources had been depleted, coverage holes in which the performance criterion could not be met would form and performance would degrade.

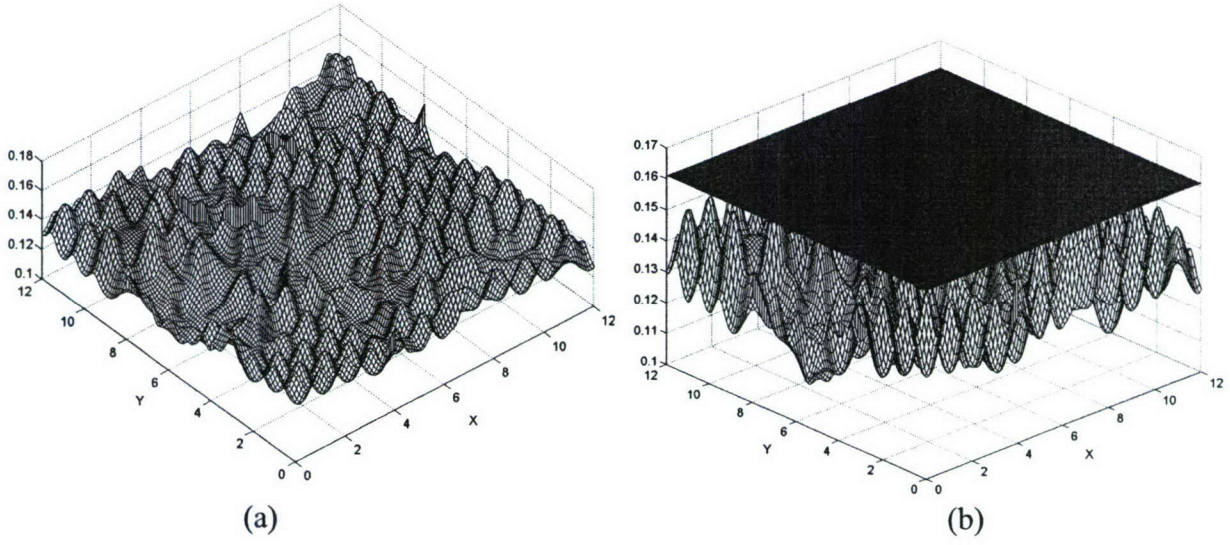


Fig. 18. (a) Computed CRLB for the RMS error in the estimate of D versus X and Y for the field at the end of the simulation at instant 1841. The simulation ended because the redundant resources in one of the clusters (cluster 8) were exhausted. (b) Computed CRLB with the performance criterion of 0.1610 for the RMS error shown as a plane. The entire coverage area met the performance criterion, which is indicated by the fact that the CRLB surface is bounded by the plane for all target positions inside the coverage area.

By simulation instant 1645, 55 sensors had been lost, with PGR used to awaken redundant sensors as needed to maintain performance at the specified criterion (Fig. 17). However, it is informative to consider how the performance would have degraded if the 55 sensors had been lost and PGR had not been used to awaken redundant sensors as needed to maintain performance. Fig. 19 shows the field layout of active sensors at instant 1645 after the death of 55 sensors in different clusters without reconfiguration to replace lost sensors. The CRLB was computed for the field without the 55 sensors that failed and without the sensors awakened using PGR. Without the sensors wakened using PGR, significant coverage holes existed for large portions of the field (Fig. 20).

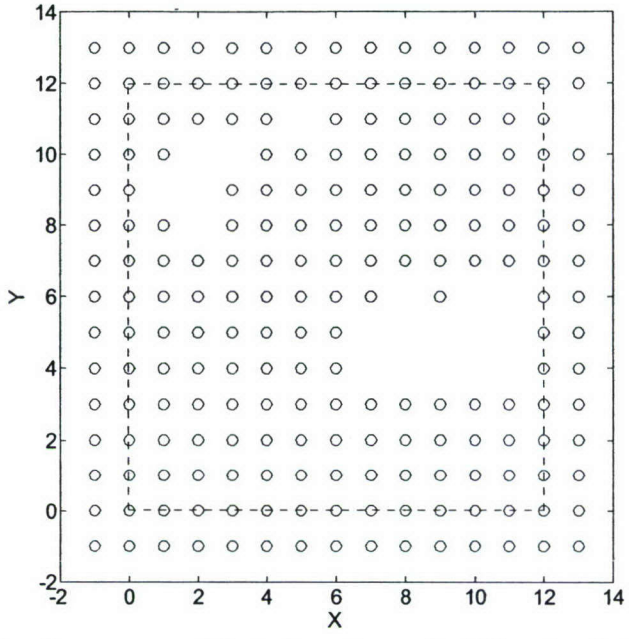


Fig. 19. Field layout of active sensors (circles) at the end of the simulation at simulation instant 1645 after the death of 55 sensors in different clusters without reconfiguration to replace lost sensors. The dotted line shows the coverage area.

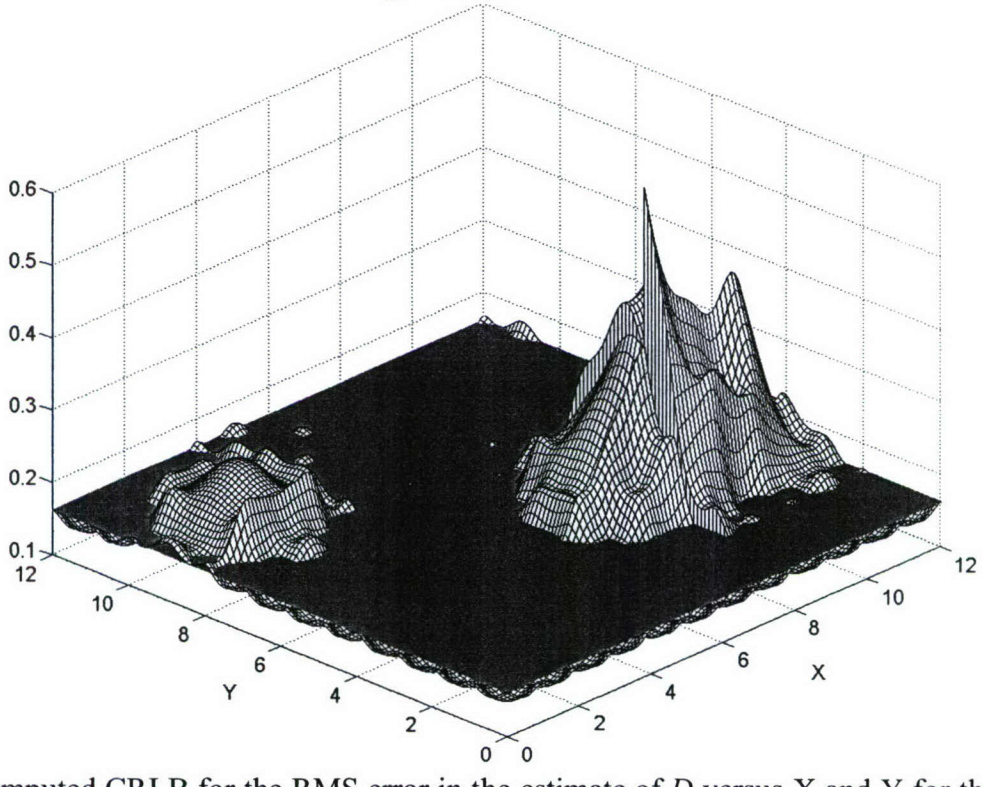


Fig. 20. Computed CRLB for the RMS error in the estimate of D versus X and Y for the field at simulation instant 1645 after the death of 55 sensors in different clusters without reconfiguration to replace lost sensors. The performance criterion of 0.1610 for the RMS error is shown as a plane. A large portion of the coverage area did not meet the performance criterion (shown by peaks above the surface of the plane).

Figures 21-30 show histograms of a) the percentage battery life remaining for each sensor in all 9 clusters and b) the number of sensors grouped by percentage battery life remaining in all 9 clusters at simulation instants 100, 200, 500, 1000, and 1500. Recall that the simulations were started with a battery life at 10 percent for all sensors in order to speed the execution of the simulation. As a consequence of the balanced power utilization achieved through sensor clustering and the PMR routing algorithm employed in the distributed system, along with the random target path of the simulation, the battery life decreased in a relatively uniform manner across clusters and across the sensors within a cluster over the time course of the simulation (Figs. 21-28). When a sensor's remaining battery life went to zero, the sensor died and redundant sensors were awakened if necessary using PGR. The first sensor died at simulation instant 1473 and the field was reconfigured using PGR. Thus, the plots at simulation instants 100, 200, 500, and 1000 reflect observations made before any sensors died, whereas the plots for simulation instant 1500 reflect observations made after some sensors had died and the field had been reconfigured. By simulation instant 1500, some sensors in clusters 7 and 8 had died, and PGR had been used to awaken some sensors in clusters 7 and 8 to maintain performance. At simulation instant 1500, the sensors in clusters 7 and 8 that had already been awakened using PGR are those sensors having a battery life at near 10 percent (Figs. 29-30).

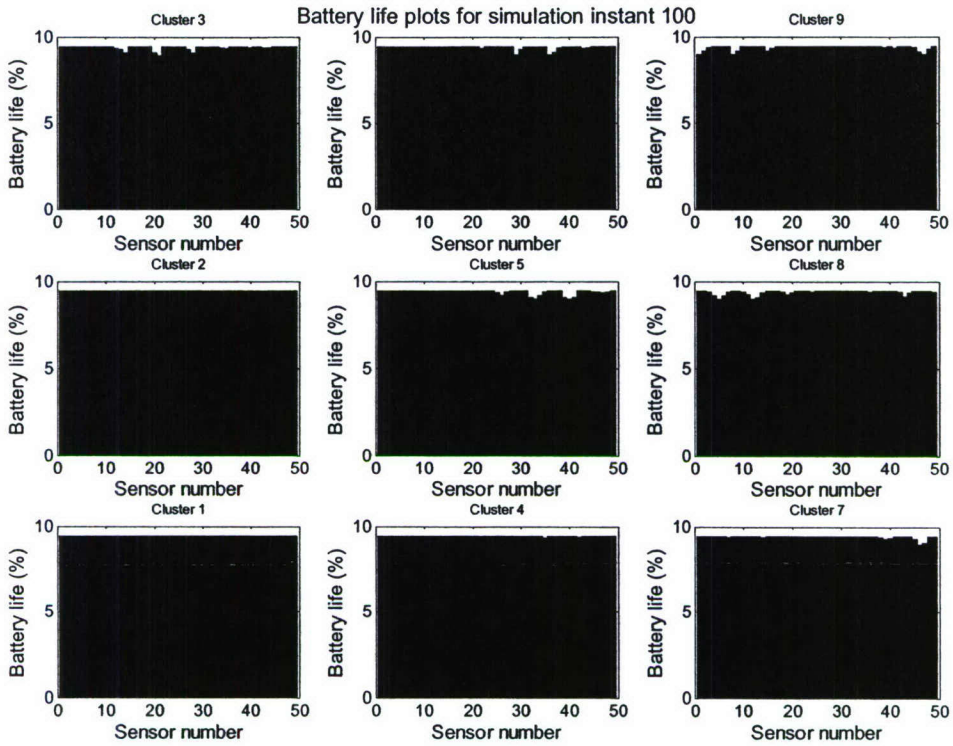


Fig. 21. Histograms of battery life remaining for the sensors within each cluster at simulation instant 100.

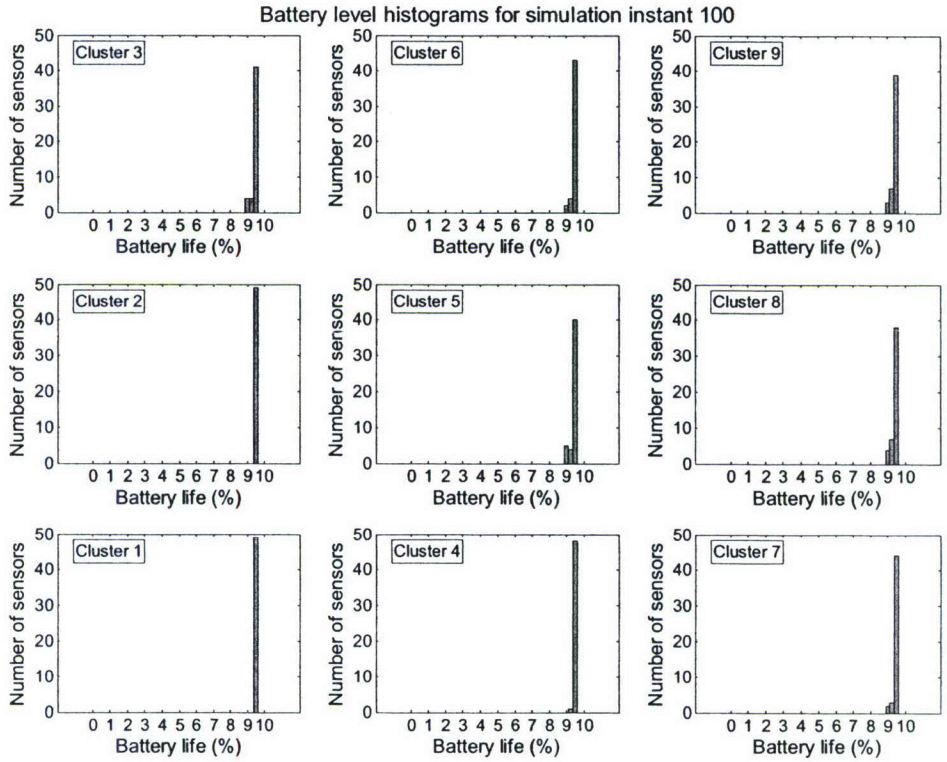


Fig. 22. Histograms of the number of sensors grouped by percentage battery life remaining within each cluster at simulation instant 100.

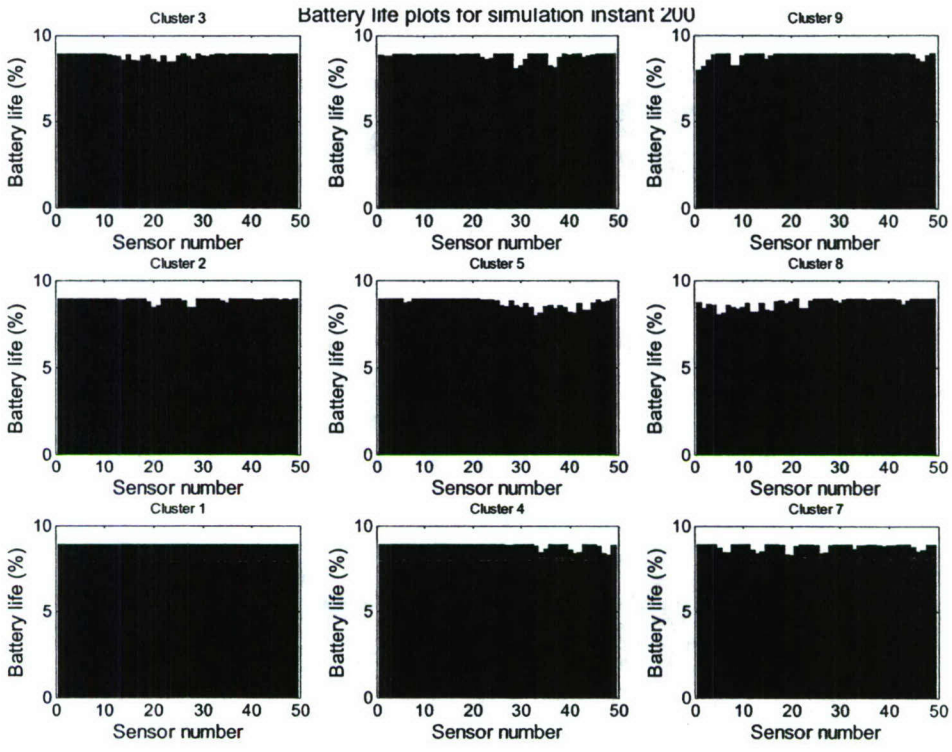


Fig. 23. Histograms of battery life remaining for the sensors within each cluster at simulation instant 200.

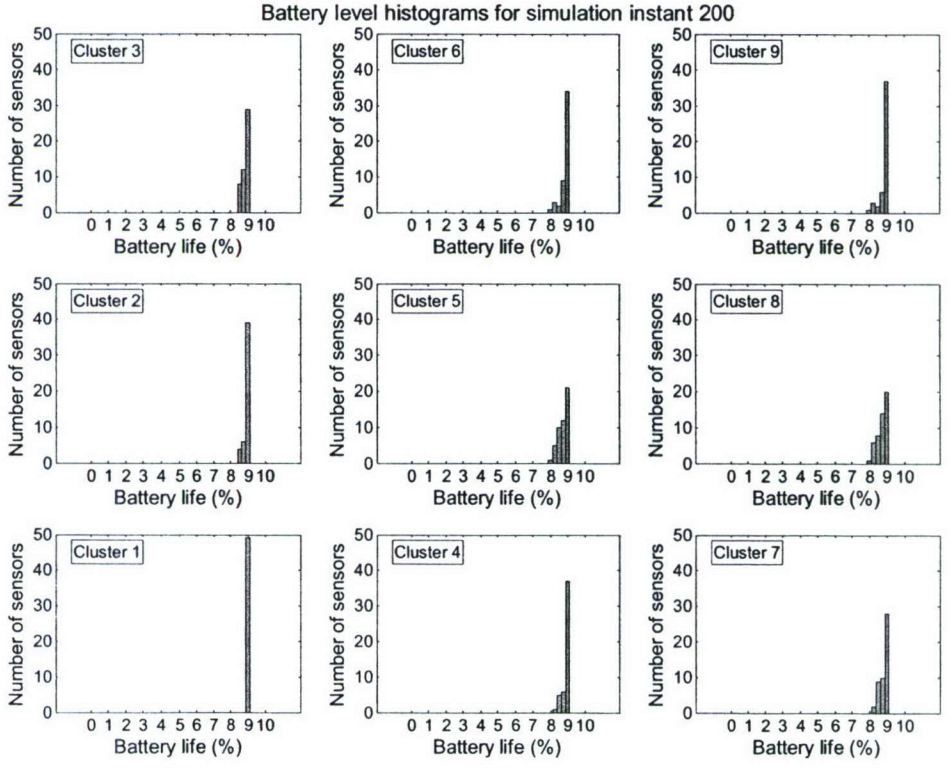


Fig. 24. Histograms of the number of sensors grouped by percentage battery life remaining within each cluster at simulation instant 200.

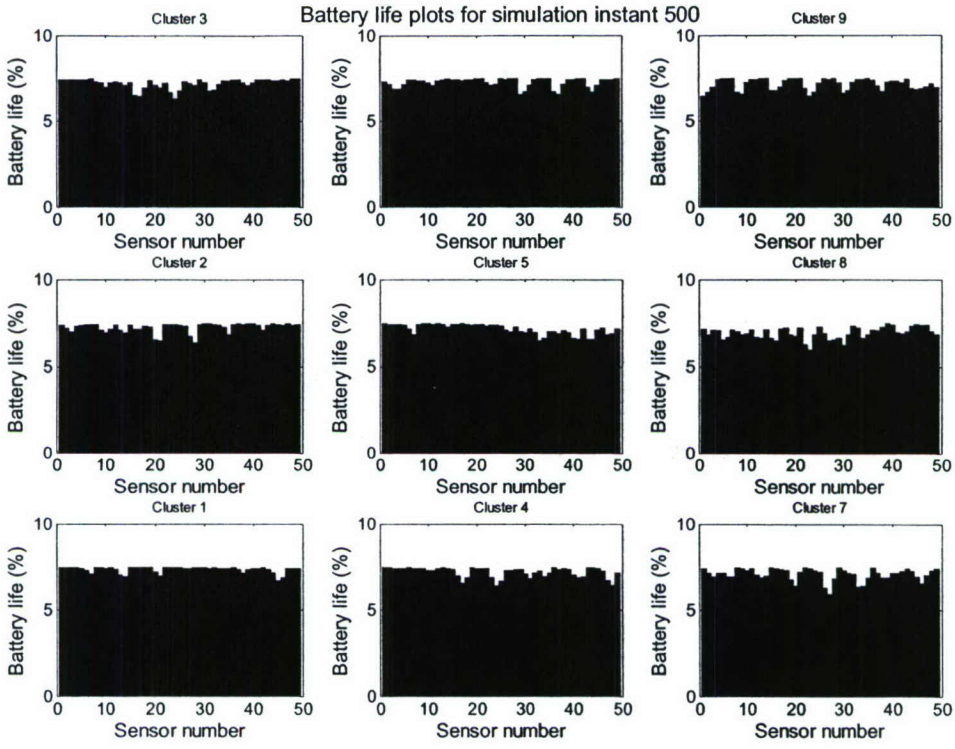


Fig. 25. Histograms of battery life remaining for the sensors within each cluster at simulation instant 500.

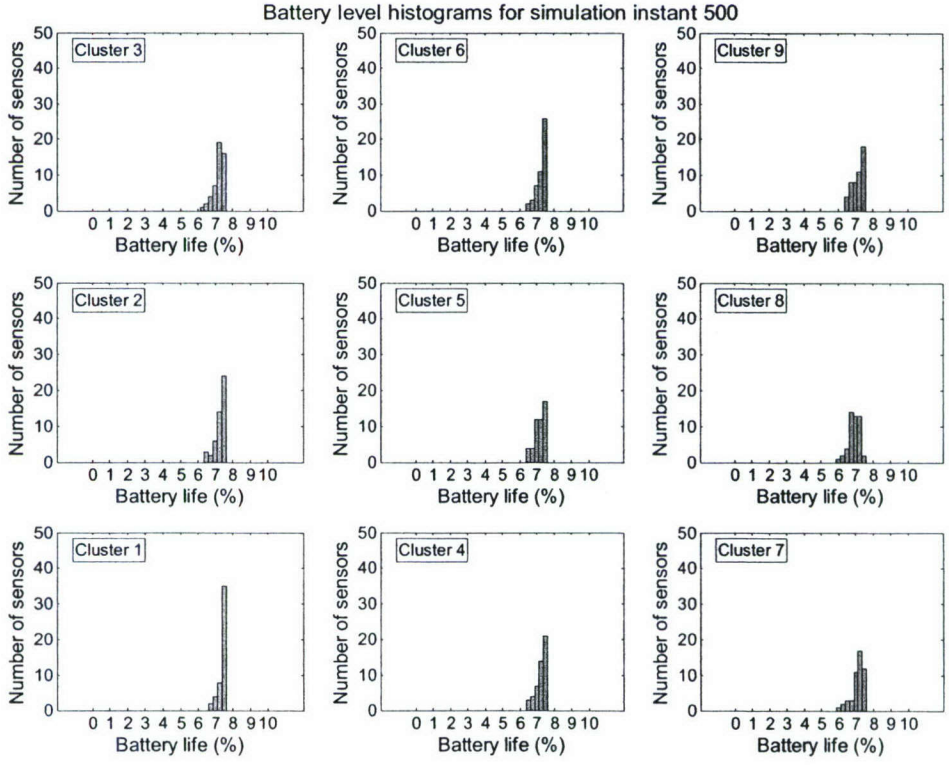


Fig. 26. Histograms of the number of sensors grouped by percentage battery life remaining within each cluster at simulation instant 500.

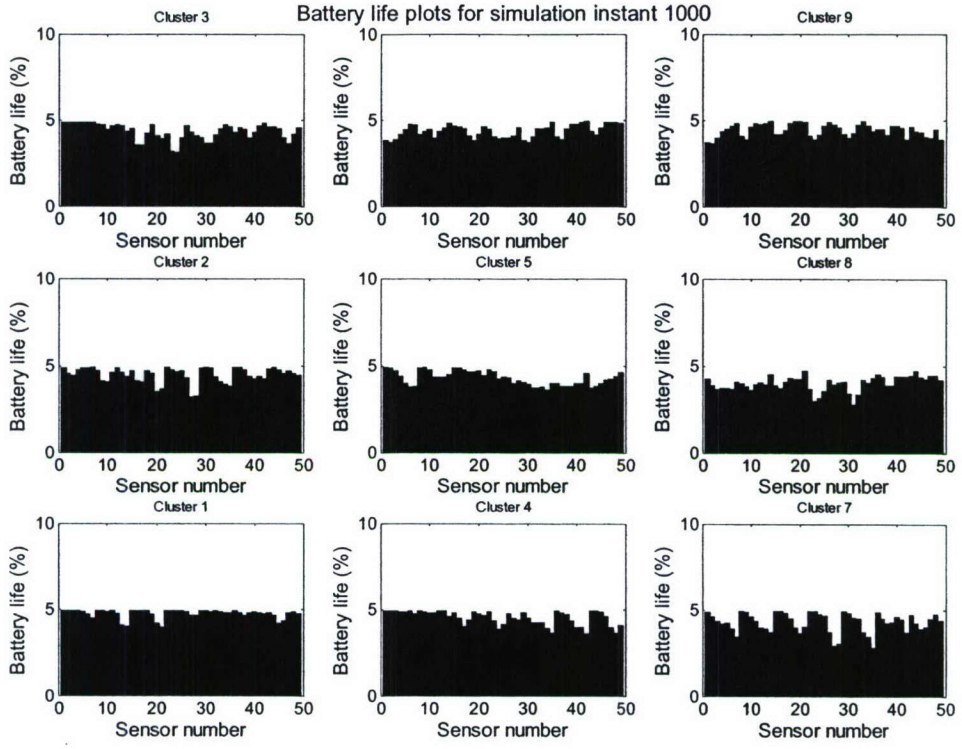


Fig. 27. Histograms of battery life remaining for the sensors within each cluster at simulation instant 1000.

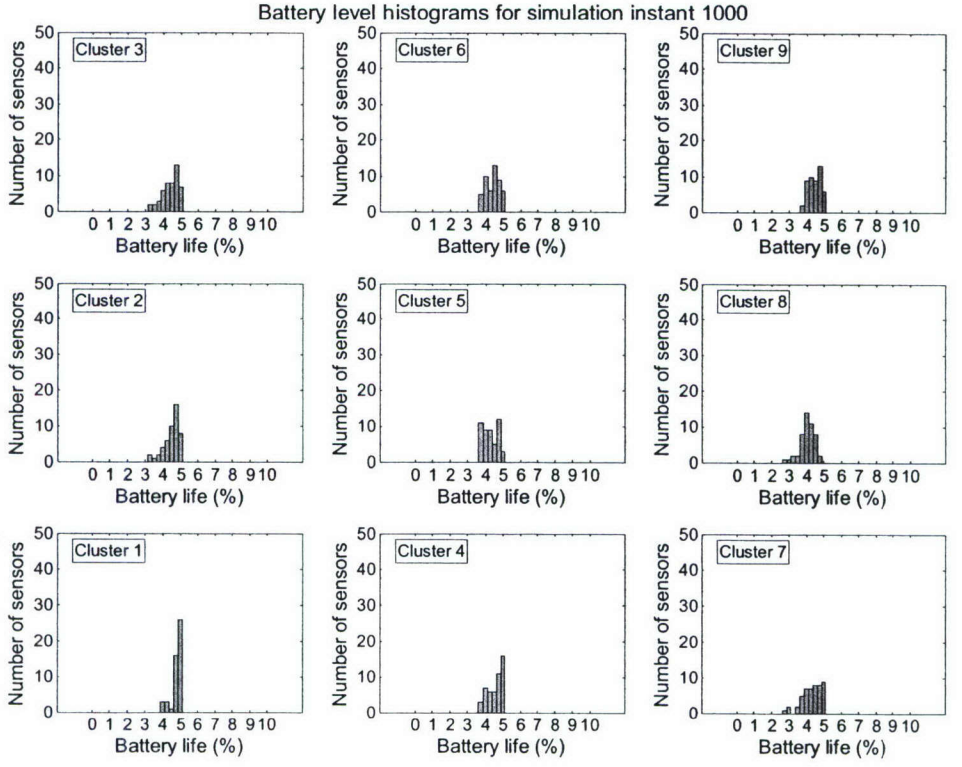


Fig. 28. Histograms of the number of sensors grouped by percentage battery life remaining within each cluster at simulation instant 1000.

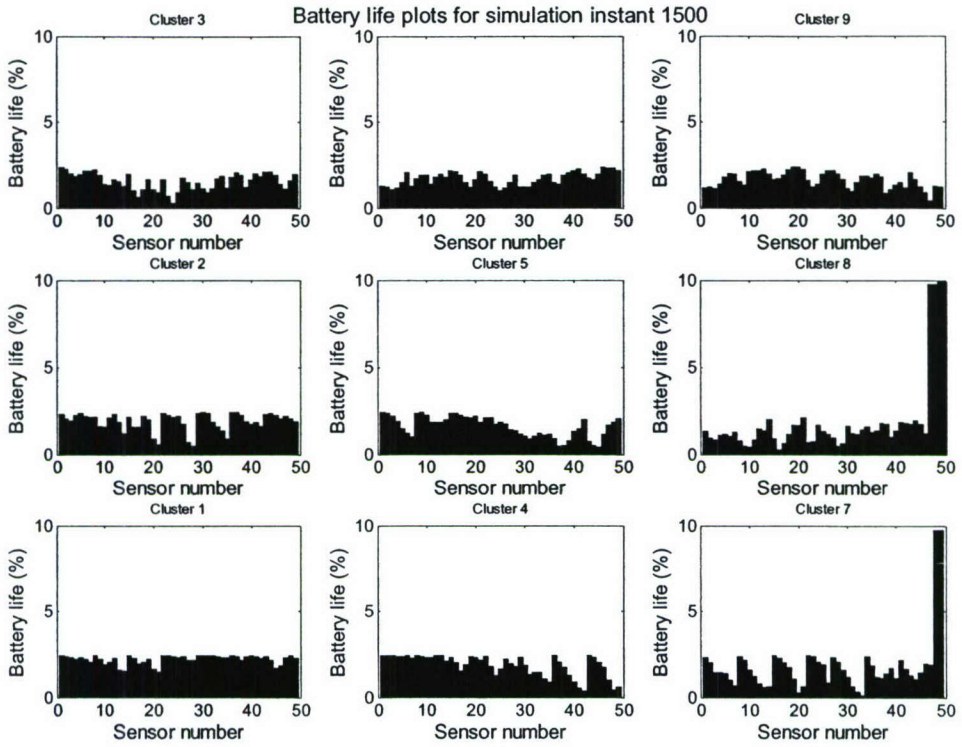


Fig. 29. Histograms of battery life remaining for the sensors within each cluster at simulation instant 1500.

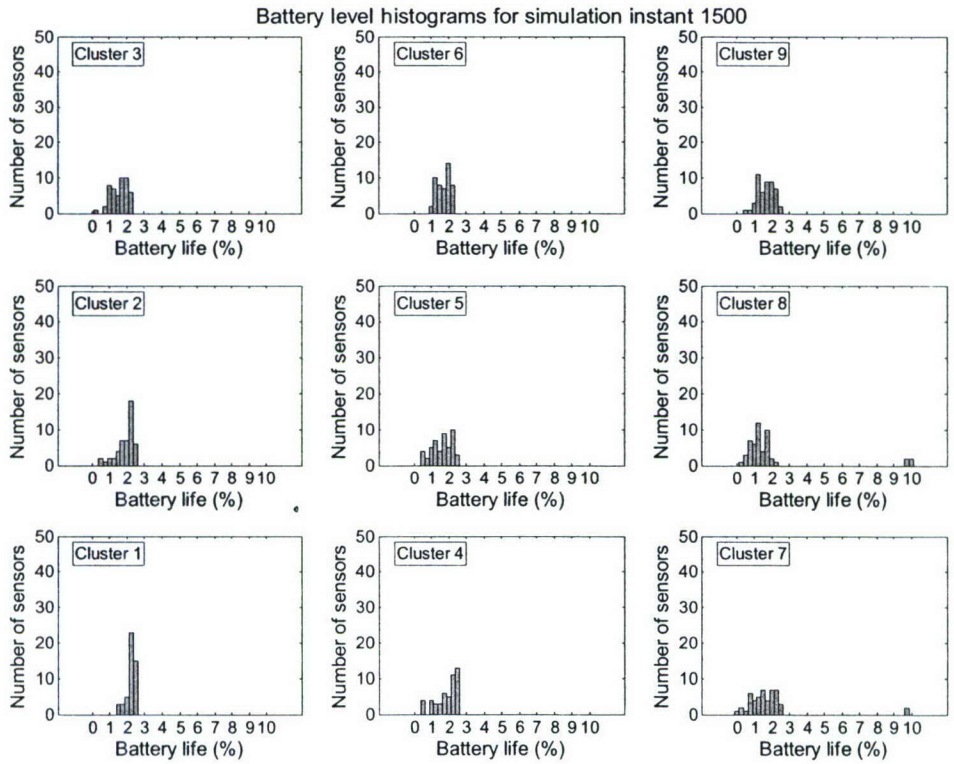


Fig. 30. Histograms of the number of sensors grouped by percentage battery life remaining within each cluster at simulation instant 1500.

VII. CONCLUSIONS

The current DADS demonstrates feasibility of an advanced concept, but does not possess a high degree of autonomy and the associated advantages, including reduced vulnerability to detection and destruction by the enemy. This research was motivated by the need to replace the centralized management employed in the current DADS with decentralized methods appropriate for the distributed Micro-DADS, and represents the state-of-the-art in distributed systems research.

The decentralized organization of Micro-DADS inherently supports survivability by facilitating the incorporation of redundancy, the balanced use of network resources, and use of local optimization that achieves global optimization. In this work, organizational and computational hierarchies for intelligent distributed systems were identified and explored. Methods for distributed detection, localization, and target tracking were developed. Strategies to allow the sensor field to survive attacks, failures, and accidents were identified and used to develop methods for optimizing performance and field life. Intelligent software agents were developed to facilitate the exchange of information, support achieving survivability, and facilitate the distribution of important system functions that include control and coordination. Computer simulations were used to evaluate algorithms and alternative approaches. Simulations were used to develop and test alternatives, consider tradeoffs, and evaluate the ability of the system to achieve its goals when presented with a set of operational challenges.

Advances in networking, microelectromechanical systems (MEMS), and sensing technologies are likely to stimulate the proliferation of distributed systems and sensing. Many of the research challenges in distributed sensing and control posed by the Micro-DADS concept are general and overlap those in other military and commercial fields. Therefore, the contributions of this research will likely affect other distributed sensing systems in military as well as commercial environments.

Research Infrastructure Development

The DEPSCoR emphasis on infrastructure development provided a significant opportunity for this grant to impact research infrastructure of the participating universities and the State of Alabama. The project contributed to high-priority efforts, support of a new program, student training, and new collaborations between faculty members having complementary expertise.

At The University of Alabama at Birmingham (UAB), the project further leveraged the resources being allocated starting in 2002 to build the area of high performance computing and simulation, an effort involving the focused recruiting of new faculty members and students and the development of the Enabling Technology Laboratory to support high performance computing, simulation, visualization, and virtual reality within the School of Engineering. New faculty collaborations and the grant's financial support for graduate assistantships stimulated the early growth of a new Computer Engineering Doctoral Program shared by UAB and the University of Alabama at Huntsville (UAH), improving the State's capacity to educate students in an area recognized as critical to national defense and future economic development of the State. Dr. Thomas C. Jannett served as project director. Dr. Wells from the University of Alabama at Huntsville considered issues related to parallel computation, performance,

simulation, and distributed reconfigurable systems. Dr. Alan Shih and Dr. Roy Koomullil of the UAB Department of Mechanical Engineering and the Enabling Technology Laboratory were primarily responsible for exploring visualization and virtual reality simulations. Each of the participating faculty members served on the committees of students supported by the research. In addition to interacting in meetings and standard methods of communication, the investigators developed and utilized Internet conferencing, with weekly Internet conferences held to link UAB and UAH participants during the middle of the project period. In addition to the faculty members supported by the budget, several faculty members were involved in the project either directly or through participation in graduate student supervisory committees.

The work provided rich educational experiences for a diverse set of graduate students in the areas of distributed systems, agents, intelligent control, simulation, and high performance computing. Students participated in the research by serving as graduate assistants and through completion of projects, theses, and dissertations. The budget included support for several Ph.D. students in each year. Partial support for M.S. students is budgeted each year to allow promising master's level students to participate in the research, gain experience, and make a more informed decision about a research career. Eleven graduate students, nine at UAB (three Ph.D. and six M.S.) and two at UAH (two Ph.D.), served as graduate assistants or took on projects related to the research over the three-year grant period. Two Computer Engineering Ph.D. students supported by the grant completed doctoral dissertations that addressed key defense-related research problems in areas related to distributed systems, simulation, and high performance computing. They have joined other universities as faculty members. Three Ph.D. students are nearing completion of their degree requirements in the areas of agents, intelligent control, and reconfigurable systems. Six students have completed the Masters Degree in Electrical Engineering (M.S.E.E.). Four are working in industry and two have been accepted in Ph.D. programs. Each of the five Ph.D. students and three M.S.E.E. students wrote and presented a conference paper. In addition, two undergraduate students became involved in the research through the departmental honors program.

VIII. REFERENCES

1. M. D. Hatch, J. L. Kaina, R. P. Mahler, R. S. Myre, "Data fusion methodologies to support theater level and deployable surveillance systems," *Conference Record of the Thirty-Second Asilomar Conf. on Signals, Systems and Computers*, 1998. pp 563 - 567, vol.1, 1-4 Nov. 1998.

2. E. Jahn, J. Kaina, and M. Hatch, "Fusion of multi-sensor information from an autonomous undersea distributed field of sensors, *Proceedings of the Second International Conference on Multisource-Multisensor Information Fusion,* Sunnyvale, California, pp. 4-11, July, 1999.

3. P. Shea and M. Owen, "Fuzzy control in the deployable autonomous distributed system," *Proceedings of SPIE: Signal Processing, Sensor Fusion, and Target Recognition VIII 1999*, volume 3720, Orlando, Florida, April, 1999.

4. D. Klamer, M. Owen, "Adaptive threshold control in an autonomous sensor field," *Proceedings of SPIE: Signal and Data Processing of Small Targets 2000*, volume 4048, Orlando, Florida, April, 2000.

5. M. W. Owen, D. M. Klamer, and B. Dean, "Evolutionary control of an autonomous field," *Proceedings of the Third International Conference on Information Fusion*, pp. MoD1-3 - MoD1-9, July 10-13, 2000.

6. Future DADS and USW Concept Briefing, Office of Naval Research Arlington, VA, May 29, 2001.

7. T. S. Lee, S. Ghosh, and A. Neerode, "Asynchronous, distributed, decision-making systems with semi-autonomous entities: a mathematical framework," *IEEE Transactions on Systems, Man and Cybernetics, Part B*, pp. 229-239, Feb. 2000.

8. J. S. Albus. "Outline for a theory of intelligence," *IEEE Trans. on Systems, Man, and Cybernetics*, pp. 473-509, May/June 1991.

9. C. Wei-Peng, J. C. Hou, and S. Lui, "Dynamic clustering for acoustic target tracking in wireless sensor networks," *IEEE Transactions on Mobile Computing*, Volume 3, Issue 3, July-Aug. 2004 Page(s):258 – 271.

10. B. Horling, R. Vincent, R. Mailler, J. Shen, K. Rawlins, and V. Lesser, "Distributed sensor network for real time tracking," *Proceedings of the Fifth International Conference on Autonomous Agents*, p.417-424, May 2001.

11. R. Brooks, P. Ramanathan, and A. Sayeed, "Distributed target classification and tracking in sensor networks," *Proc. IEEE*, vol. 91, no. 8, pp. 1163–1171, Aug. 2003.

12. R. Brooks, C. Griffin, and D. S. Friedlander, "Self organized distributed sensor network entity tracking (special issue on Sensor Networks)," *Internat. J. High Performance Comput. Appl.* 16 (3) (2002) 207-220.

13. R. Quintero and A. J. Barbera, "A real-time control systems methodology for developing intelligent control systems," US Dept. of Commerce, 1992, NISTIR 4936.

14. G. N. Saridis, "Analytic formulation of the principle of increasing precision with decreasing intelligence for intelligent machines," *Automatica*, vol. 25, pp. 461-467, 1989.

15. T. S. Perraju, "An agent framework for survivable network systems," *IEEE International Performance, Computing and Communications Conference*, pp. 469-475, 10-12 Feb. 1999.

16. T. S. Perraju, "Agents and autonomous distributed systems," *Proceedings of the Fourth International Symposium on Autonomous Decentralized Systems*, 1999. pp. 264-266, 21-23 March 1999.

17. D. Klamer, M. Moore, P. K. Varshney, and R. Niu, "Decision fusion in a wireless sensor network with a large number of sensors," *Proceedings of the Seventh International Conference on Information Fusion*, Stockholm, Sweden, June 2004.

18. R. Niu and P. K. Varshney, "Distributed detection and fusion in a large wireless sensor network of random size," *EURASIP Journal on Wireless Communications and Networking* 2005:4, 462–472.

19. R. Niu and P. K. Varshney, "Target location estimation in wireless sensor networks using binary data," *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, May 2001, pp. 2037-2040.

20. R. Niu and P. K. Varshney, "Target location estimation in sensor networks with quantized data," *IEEE Transactions on Signal Processing*, vol. 54, pp. 4519-4528, December 2006.

21. M. M. Noel, "Explorations in swarm algorithms: Hybrid particle swarm optimization and adaptive culture model algorithms," PhD Dissertation, The University of Alabama at Birmingham, 2005.

22. M. M. Noel and T. C. Jannett, "A new continuous optimization algorithm based on sociological models," *Proceedings of the 2005 American Control Conference*, 2005. Pages: 237 - 242. (file: Noel_ACC_2005).

23. M. M. Noel, P. P. Joshi, and T. C. Jannett, "Improved maximum likelihood estimation of target position in wireless sensor networks using particle swarm optimization," *Proceedings of the Third International Conference on Information Technology: New Generations (ITNG'06)*, 2006, Pages: 274-278. (file: Noel_ITNG_2006).

24. S.T. Gurumani, M. M. Noel, E. B. Wells, and T. C. Jannett, "Performance analysis of coarse-grained parallel particle swarm optimization," *Proceedings of the ISCA 19th International Conference on Parallel and Distributed Computing Systems*, 2006, Pages: 197-202. (file: Gurumani_CPDCS_2006).

25. P. Joshi and T. C. Jannett, "A simple two-step approach for maximum likelihood target localization using binary data," submitted to ICWN'07- The 2007 International Conference on Wireless Networks. (file: Joshi_ICWN_2007).

26. J. L. Elliott, M. M. Noel, P. P. Joshi, and T. C. Jannett, "Simulating the effect of uncertainty in sensor positions on the accuracy of target localization in wireless sensor networks," *Proceedings of IEEE Southeastcon 2006*, 2006. Pages: 120-124. (file: Elliott_SECON_2006).

27. R. Kasthurirangan, "Comparison of architectures for multisensor data fusion in deployable autonomous distributed systems," Master's thesis, Department of Electrical and Computer Engineering, The University of Alabama at Birmingham, 2003.

28. H. Hashemipour, S. Roy, and A. Laub, "Decentralized structures for parallel Kalman filtering," *IEEE Transactions on Automatic Control*, vol. 33, Jan. 1988.

29. B.S. Rao and H.F. Durrant-Whyte, "Fully decentralised algorithm for multisensor Kalman filtering," *IEE Proceedings-Control Theory and Applications*, pp. 413-420, Sept. 1991.

30. J. Manyika and H. Durrant-White, Data Fusion and Sensor Management: A Decentralized Information-Theoretic Approach, Prentice Hall, December 1994.

31. A. G. O. Mutambara, Decentralized Estimation and Control for Multi-Sensor Systems, CRC Press, 1998.

32. A. Anthony and T. C. Jannett, "A framework for using agents in distributed sensor networks," *Proceedings of IEEE Southeastcon 2006*, 2006. Page: 337. (file: Anthony_SECON_2006).

33. S. V. Chandrachood, A. Anthony, and T. C. Jannett, "Using resource based modeling to evaluate coordination schemes in wireless sensor networks," *Proceedings of IEEE Southeastcon 2006*, 2006. Pages: 91-97. (file: Chandrachood_SECON_2006).

34. A. Anthony and T. C. Jannett, "Measuring machine intelligence of an agent-based distributed sensor network system," *Proceedings of the International Joint Conferences on Computer, Information, and Systems Sciences, and Engineering (CSSE 2006)*, 2006. (file: Anthony_CSSE_2006).

35. A. Anthony and T. C. Jannett, "Stacking functional agents in sensor network systems," submitted to IEEE Southeastcon 2007, 2007. (file: Anthony_SECON_2007).

36. A. Anthony and T. C. Jannett, "Comparing agent paradigms for resource management in sensor networks," submitted to ICWN'07- The 2007 International Conference on Wireless Networks. (file: Anthony_ICWN_2007).

37. H. Yang and B. Sikdar, "A protocol for tracking mobile targets using sensor networks," *Proceedings of the First IEEE International Workshop on Sensor Network Protocols and Applications,* May, 11, 2003, pp. 71-81.

38. Y. Tashtoush, B. E. Wells, and T. C. Jannett, "Applying fuzzy-reinforcement learning to track a mobile target using a wireless sensor network," *Proceedings of the 2005 International Conference on Wireless Networks (ICWN'05)*, Editors: L. T. Yang, H. R. Arabnia, and L-C. Wang, June 2005. Pages: 427-433. (file: Tashtoush_ICWN_2005).

39. R. Praveenkumar and T. C. Jannett, "Investigation of routing protocols in a sensor network using resource based models," submitted to IEEE Southeastcon 2007, 2007. (file: Praveen_SECON_2007).

40. M. Gaitonde, "Exploring decision algorithms for optimization in networked sensor systems," Master's thesis, Department of Electrical and Computer Engineering, The University of Alabama at Birmingham, 2005.

41. K. Chakrabarty, S. S. Lyengar, H. Qi, and E. Cho, "Grid coverage for surveillance and target location in distributed sensor networks," *IEEE Trans. on Computers,* vol. 51, no. 12, Dec. 2002, pp. 1448-1453.

42. P. P. Joshi and T. C. Jannett, "Performance-guided reconfiguration of wireless sensor networks that use binary data for target localization," *Proceedings of the Third International Conference on Information Technology: New Generations (ITNG'06)*, 2006, Pages: 562-565. (file: Joshi_ITNG_2006).

43. P. Joshi and T. C. Jannett, "Simulation of localized performance-guided reconfiguration of distributed sensor networks," submitted to Third International Conference on Information Technology: New Generations (ITNG'07), 2007. (file: Joshi_ITNG_2007).

44. P. Joshi and T. C. Jannett, "Localized performance-guided reconfiguration for distributed sensor networks," submitted to IEEE Southeastcon 2007, 2007. (file: Joshi_SECON_2007).

45. V. Lesser, C. L. Ortiz, and M. Tambe, Distributed Sensor Networks: A Multiagent Perspective, Springer, 2006.

46. A. L. Rajendra and T. C. Jannett, "Comparison of glider routing algorithms for data acquisition in undersea sensor networks," submitted to IEEE Southeastcon 2007, 2007. (file: Rajendra_SECON_07).

47. T. C. Jannett, A. M. Shih, B. E. Wells, A. Anthony, S. V. Chandrachood, P. Joshi, R. Kasthurirangan, M. M. Noel, D. Vetrivel, and J. Winningham, "Simulation and visualization in the development of an underwater distributed sensor field," *Proceedings of the Huntsville Simulation Conference*, Huntsville, Alabama, October 19-21, 2004. (file: Jannett_HSC_2004).

APPENDICES

1. P. P. Joshi and T. C. Jannett, "Repeated Trials for Target Detection in a Hierarchical Sensor Network Employing a Large Number of Sensors," Technical Report, The University of Alabama at Birmingham, 2007

2. P. P. Joshi and T. C. Jannett, "A simple two-step approach for maximum likelihood target localization using binary data," submitted to ICWN'07- The 2007 International Conference on Wireless Networks.

3. A. Anthony and T. C. Jannett, "Comparing agent paradigms for resource management in sensor networks," submitted to ICWN'07- The 2007 International Conference on Wireless Networks.

4. P. P. Joshi and T. C. Jannett, "Localized performance-guided reconfiguration for distributed sensor networks," submitted to IEEE Southeastcon 2007, 2007.

Repeated Trials for Target Detection in a Hierarchical Sensor Network Employing a Large Number of Sensors

Parag P. Joshi and Thomas C. Jannett
Department of Electrical and Computer Engineering
The University of Alabama at Birmingham
Birmingham, Alabama, USA 35294

I. Introduction

Sensor networks employing a large number of relatively inexpensive sensors having a limited detection range and communication capability are an area of significant research interest [1]. For a network having a large number of sensors, a fusion center can make a final decision about a target's presence or absence with a decision fusion rule that uses the total number of detections reported by local sensors for hypothesis testing. However, a very large number of sensors may be needed to achieve a high probability of detection (PD) and low probability of false alarm (PF) performance [1]. Centralized methods such as this suffer from weaknesses including a high vulnerability to failure and a lack of scalability. If the central entity or any of its key components fails, the system cannot function. As the system complexity and size increase, the system becomes limited by the processing and communication capacities of the central manager.

This report considers distributed data fusion within multi-tier hierarchical sensor network architectures. Employing repeated trials at different tiers facilitates achieving specified probability of detection (PD) and probability of false alarm (PF) performances using inexpensive sensors having a limited detection range and communications capability.

The Three-Tier Hierarchical Decision Structure

A hierarchical sensor network (field) is functionally organized into multiple layers. Entities at the different network layers are responsible for reporting data to those at higher layers. A three-layer generalization of a hierarchical sensor network is shown in Fig. 1. The figure and its description can be extended to apply to other multi-tier sensor networks having more than three tiers.

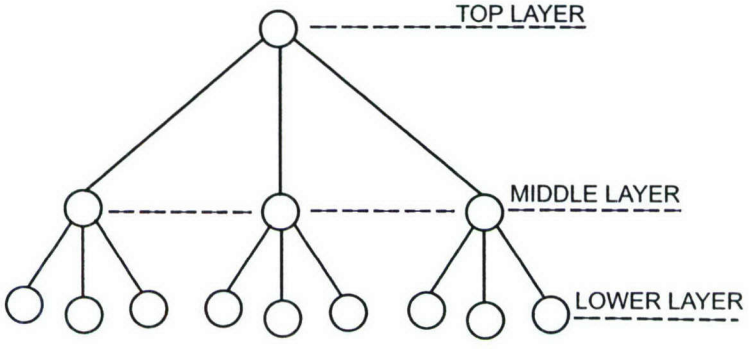


Fig. 1. The three-tier hierarchical decision structure

In this work, the function of the hierarchical sensor network is to detect the presence of a target in a geographical area. The nodes (common sensors) at the lowest network layer are responsible for sensing the presence of a target in their vicinity. Each sensor makes a decision about a target's presence by comparing the intensity of a measured signal to a threshold. The sensor node reports its decision to a middle layer node.

The middle layer nodes are responsible for fusing the data received from the common sensors. The middle layer node applies an appropriate fusion technique (decision fusion or value fusion) to decide about the presence of a target within the group of sensors that reports to it [2]. Each middle layer node transmits its decision to the top layer node. The top layer node fuses data from all such middle layer nodes in the field to make a decision about the presence of a target within the field. In this work, the middle layer node is referred to as a gateway node and the group of sensors that report to a gateway node are referred to as the gateway neighborhood. The gateways represent the paths through which low-level detections must pass through the network in order to produce a detection at the field level.

The organization of the rest of this report follows. Section II considers two-tier sensor networks and reviews system performance measures based on the probability of detection (PD) and probability of false alarm (PF) presented in [1]. The two-tier approach is useful to design sensor networks with a large number of low-level sensors, but does not present guidelines to design hierarchical sensor networks that have more than two layers. In addition, the approach does not provide sufficient degrees of freedom necessary to obtain the desired level of performance (detection probability and false alarm rate). Section III extends the work presented in [1] for networks with more than two tiers, and introduces the use of repeated trials to achieve a desired performance. Section IV discusses the field level false alarm performance, and derives design parameters useful in attaining a specified field level false alarm performance. Section V describes the procedures applied to demonstrate the use of repeated trials to attain the degrees of freedom needed to achieve a desired level of performance using multiple non-overlapping gateway areas in a hierarchical network. Section VI presents the results, section VII gives the discussion, and section VIII presents the conclusions.

II. Performance Measures for a Two-Tier Decision Structure

The approach in [1] described a two-tier sensor network (or gateway neighborhood) to monitor an area using a large number of sensors randomly deployed over the region, and gave probabilistic measures of performance in terms of detection and false alarm probabilities. The work provided a necessary building block to test the usefulness of repeated trials in hierarchical sensor networks that could have more than two tiers. This section reviews the work in [1] describing two-tier sensor networks and corresponding performance measures applied to a single gateway neighborhood in which a group of sensor nodes report to a fusion node that makes a detection decision.

The Binary Hypothesis Testing Approach for Decision Making

Binary hypothesis testing is a special case of a detection problem in which the decision space consists of only two possibilities (δ_0 and δ_1) and there is a hypothesis corresponding to each decision. For the problem considered in this work, the null hypothesis is H_0, and the alternative hypothesis is H_1. For the target detection problem, H_0 denotes the event that a target is absent, and H_1 denotes the event that a target is present.

The binary hypothesis-testing problem has four possible outcomes:

1. H_0 was true, δ_0 was chosen: Correct decision
2. H_1 was true, δ_1 was chosen: Correct decision
3. H_0 was true, δ_1 was chosen: False alarm (also known as Type I error)
4. H_1 was true, δ_0 was chosen: Missed Detection (also known as Type II error)

Our aim is to be able to specify the rate of false alarms that occur at the field level, and design the field accordingly.

Gateway Area Sensor Placement

As shown in Fig. 2 below, a total of N sensors are randomly placed in the region of interest (ROI), a gateway area that is a square of area a^2. The target's location in ROI is uniformly random. The locations of sensors are independent with sensors identically distributed (iid) and distributed uniformly in the ROI as follows.

$$f(x_i, y_i) = \frac{1}{a^2}(-\frac{a}{2} \le x_i, y_i \le \frac{a}{2}) \tag{1}$$

for $i = 1 \ldots N$, where (x_i, y_i) are the coordinates of sensor i.

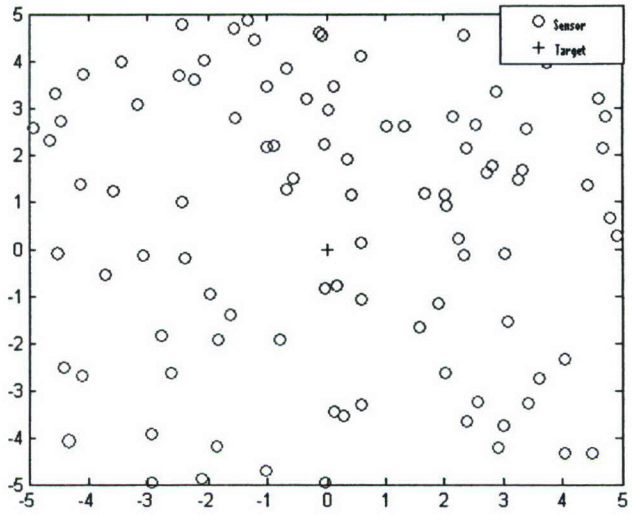


Fig. 2. Region of interest (ROI), which is a neighborhood of area a^2. Sensor placement is uniformly random. The target is placed at the center of the ROI.

For a common sensor i in a gateway neighborhood, the hypothesis testing problem is

$$H_1: s_i = a_i + n_i$$

$$H_0: s_i = n_i$$

where a_i, the power of the signal received, is determined using an isotropic energy model

$$a_i = \sqrt{\frac{P_o}{1+\alpha d_i^{\,n}}}. \tag{2}$$

In (2), P_o is the signal power emitted by the target at a distance zero from the sensor, α is an adjustable constant (chosen to be 200), and n is the signal decay constant (chosen to be 2 or 3). The distance between a target located at (x_t, y_t), and local sensor i located at (x_i, y_i) is

$$d_i = \sqrt{(x_i - x_t)^2 + (y_i - y_t)^2}. \tag{3}$$

Noise at a local sensor i follows the standard Gaussian distribution $n_i \approx N(0,1)$.

Sensor Level Performance

Each sensor makes a detection decision if the noisy signal, s_i, exceeds a threshold. The probability of detection at the sensor level is denoted as *pd*, and the probability of false alarm is denoted as *pfa*. For every sensor, *pfa* is a function of the local threshold alone, and for a common threshold, τ, it is

$$pfa = \frac{1}{\sqrt{2\pi}} \int_{\tau}^{\infty} e^{-\frac{t^2}{2}} dt. \tag{4}$$

Similarly, for each sensor *pd* is calculated as

$$pd = \frac{1}{\sqrt{2\pi}} \int_{\tau}^{\infty} e^{\frac{-(t-a_i)^2}{2}} dt. \tag{5}$$

Thus, *pfa* is set by τ and is the same for all sensors, but *pd* differs from one sensor to another since it is a function of d_i.

Gateway Level Fusion Rule

A gateway decides that there is a target in its neighborhood based on the number of sensors reporting 1's as a statistic. It uses a threshold, *TG*, for making a decision

$$\Lambda = \sum_{i=1}^{N} I_i \underset{H_0}{\overset{H_1}{>}} TG \tag{6}$$

where I_i denotes the binary decision from a local sensor *i*, i.e., $I_i = \{0, 1\}$. I_i takes the value 1 when the sensor decides that a target is present; otherwise, it takes a value of 0. The probability of false alarm at the gateway level, P_f, is

$$P_f = Pr\{\Lambda = \sum_{i=1}^{N} I_i \geq T \mid H_0\}. \tag{7}$$

Under hypothesis H_0, the total number of detections $\Lambda = \sum_{i=1}^{N} I_i$ follows a binomial (N, pfa) distribution. Therefore, for a given gateway threshold *TG*, P_f is calculated as

$$P_f = \sum_{i=TG}^{N} \binom{N}{i} \cdot p_{fa}{}^{i} \cdot (1-p_{fa})^{N-i}. \tag{8}$$

Since *pd* differs from one sensor to another, the detections under hypothesis H_1 do not follow a binomial (N, pd) distribution, and therefore an expression for the probability of detection for the gateway, P_d, is hard to derive analytically. Hence, the Central Limit Theorem [1] is used to approximate P_d as follows.

$$\overline{p}_d = \frac{2\pi}{a^2} \int_0^{\frac{a}{2}} Q\left(\tau - \sqrt{\frac{P_0}{1+\alpha \cdot r^n}}\right) r \cdot dr + \left(1 - \frac{\pi}{4}\right) \cdot p_{fa} \tag{9}$$

$$\overline{\sigma^2} = \int_0^{\frac{a}{2}} \left(1 - Q\left(\tau - \sqrt{\frac{P_0}{1+\alpha \cdot r^n}}\right)\right) \cdot Q\left(\tau - \sqrt{\frac{P_0}{1+\alpha \cdot r^n}}\right) \cdot r \cdot dr + \left(1 - \frac{\pi}{4}\right) \cdot p_{fa} \cdot \left(1 - p_{fa}\right) \tag{10}$$

The use of polar coordinates in (9) and (10) is explained in [1]. The gateway level PD, P_d, is calculated by substituting (9) and (10) in (11) below.

$$P_d = Q\left(\frac{T - N \cdot \overline{p}_d}{\sqrt{N \cdot \overline{\sigma^2}}}\right) \tag{11}$$

Equations 9 and 10 cannot be solved without using the Q function, defined in [1] as the complementary distribution function of the standard Gaussian

$$Q(x) = \int_x^\infty \frac{1}{\sqrt{2\pi}} e^{-\frac{t^2}{2}} dt. \tag{12}$$

The integrals in equations (9) and (10) have been approximated.

The Q function is related to the complementary error function of x, *erfc(x)* as follows.

$$erfc(x) = \frac{2}{\sqrt{\pi}} \int_x^\infty e^{-\frac{t^2}{2}} dt$$

Setting $t = \frac{u}{\sqrt{2}}$ implies that $dt = \frac{du}{\sqrt{2}}$ with

$$\begin{aligned} erfc(x) &= \frac{2}{\sqrt{\pi}} \int_{\sqrt{2}x}^\infty e^{-\frac{u^2}{2}} du \cdot \frac{1}{\sqrt{2}} \\ &= 2\int_{\sqrt{2}x}^\infty \frac{1}{\sqrt{2\pi}} e^{-\frac{u^2}{2}} du \\ &= 2 \cdot Q\left(\sqrt{2}x\right) \end{aligned}$$

$$\therefore Q(x) = \frac{1}{2} erfc\left(\frac{x}{\sqrt{2}}\right). \tag{13}$$

Equation (13) can be used to compute the integrals in (9) and (10) and to compute P_d by approximation with (11). A plot of P_f versus P_d for different values of the gateway threshold is called the receiver operating characteristic (ROC). The ROC is useful tool for evaluating the performance of the gateway and for comparing theoretical results with simulations.

III. Improving the Performance of Hierarchical Networks through Repeated Trials

The approach described in the preceding section is useful to design two-tier sensor networks [1], but does not provide sufficient degrees of freedom to obtain the desired level of performance and does not consider hierarchical sensor networks that have more than two layers. The work in [3] proposes a hierarchical decision structure, but does not provide sufficient degrees of freedom to obtain the desired level of performance at every network layer if the common sensors have a limited performance. This section proposes the method of repeated trials to achieve improved performance at every layer of the multi-tier hierarchical network.

Repeated Trials

The following explains the process of repeated trials. For a gateway threshold *TG,* a target is declared present if the number of detecting sensors is greater than or equal to *TG*. A decision made using (6) based on the comparison of the number of sensor detections to *TG* constitutes one trial at the gateway level. However, results show that good detection and false alarm performances are hard to achieve with poor sensors unless the number of sensors is very high [1]. A process of repeated trials is used to improve the performance at the gateway level and field levels in the network. In order to improve the PD and PF at the gateway level, n such trials are done, and the target is declared present only if the number of sensor detections meets or exceeds the threshold *TG*, in say, T or more trials out of the n trials. By repeating trials, a higher PD and lower PF are achieved. The same process of repeated trials can be done at the field level to achieve a higher PD and lower PF at the field level.

The process of repeated trials affords better performance, and in addition, grants more degrees of freedom (the new threshold and number of repeated trials) to use when designing the network. The process can be easily understood from the following expression, which is used to compute the new performances at the gateway level and fusion center level [4].

$$P_n = \sum_{k=T}^{n} \binom{n}{k} p^k (1-p)^{n-k} \tag{14}$$

In (14), p denotes the probability of the event in a single trial, P_n denotes the probability achieved through repeated trials, T is the new threshold applied, and n is the number of times that the trial is repeated.

Gateway Level Performance with Repeated Trials

If a decision (6) about the presence of a target in a gateway neighborhood is made solely based upon a single set of binary decisions from all the sensors in the gateway neighborhood, the results are usually not satisfactory (P_d is low, and P_f is high). Repeating trials at the gateway level can result in better performance. For repeated trials, a new threshold, T_g, is introduced that requires that the threshold *TG* be satisfied in (6) at least $k = T_g$ times out of the n times the gateway trial is repeated.

For repeated trials, using (14) the gateway level detection and false alarm probabilities are given by

$$PD_g = \sum_{k=T_g}^{n} \binom{n}{k} P_d^{\,k} (1-P_d)^{n-k} \tag{15}$$

$$PF_g = \sum_{k=T_g}^{n} \binom{n}{k} P_f^{\,k} (1-P_f)^{n-k} \tag{16}$$

where P_d and P_f are the probabilities of detection and false alarm with a single set of binary decisions from the sensors in the gateway neighborhood.

From (15) and (16), the gateway level probabilities of detection and false alarm are dictated by the choice of T_g and the previously determined P_d and P_f. Since the number of repeated trials, n, can be adjusted along with the new gateway level threshold, T_g, there are more degrees of freedom in controlling gateway level performance (PD_g and PF_g) than there were for controlling P_d and P_f. This first level of repeated trials is representative of the performance (PD_g and PF_g) expected from a single gateway area if repeated trials are done, instead of using a single gateway scan (6), to detect the target.

Additional Levels of Repeated Trials

In order to achieve more degrees of freedom in controlling performance (PD and PF), trials can be repeated as many times as is necessary at any tier within the hierarchical network to improve the performance. The target is assumed to be stationary for the entire duration of the trials.

Field Level Performance

After every gateway carries out repeated trials, it reports *nfg* decisions to the fusion center in a single epoch. For each gateway, the fusion center treats this report as a repeated trial. If the number of detections reported for any gateway meets or exceeds the threshold $k = T_{field}$ times out of the *nfg* trials, the fusion center declares a target detection in the gateway area that reports the highest number of detections in the time epoch. As described in [3], the field level probability of detection is

$$PD_{field} = \sum_{k=T_{field}}^{nfg} \left\{ \binom{nfg}{k} PD_g^{k} (1-PD_g)^{nfg-k} \cdot \left[\sum_{j=0}^{k-1} \binom{nfg}{j} PF_g^{\,j} (1-PF_g)^{nfg-j} \right]^{ng-1} \right\} \tag{17}$$

where T_{field} is the field level threshold, *nfg* is the number of gateway reports in one field level epoch, *ng* is the number of gateways in the field, and PD_g and PF_g are the probabilities of detection and false alarm achieved at the gateway level.

A field level false alarm occurs when the target is absent and one or more gateways in the field have T_{field} or greater detections out of the *nfg* decisions reported to the fusion center. From (16), the probability that a gateway has equal to or more than T_{field} false alarms out of *nfg* decisions sent to the fusion center is

$$PF_g^* = \sum_{m=T_{field}}^{nfg} \binom{nfg}{m} (PF_g)^m (1-PF_g)^{nfg-m} \tag{18}$$

The field level probability of false alarm is

$$PF_{field} = \sum_{k=1}^{ng} \binom{ng}{k} (PF_g^*)^k (1-PF_g^*)^{ng-k} \tag{19}$$

Equation (19) can also be expressed as

$$\begin{aligned} PF_{field} &= \Pr(\text{False alarm at least one gateway}) \\ &= 1 - \Pr(\text{False alarm at no gateway}) \end{aligned}$$

which can be rewritten as

$$PF_{field} = 1 - \sum_{k=0}^{0} \binom{ng}{k} (PF_g^*)^k (1-PF_g^*)^{ng-k} \tag{20}$$

$$\therefore PF_{field} = 1-(1-PF_g^*)^{ng}.$$

If the *nfg* decisions that were reported to the fusion center were used to carry out a gateway level detection using *nfg* repeated trials instead of being used to make a decision at the field level, the probability of detection PD_g^* for the gateway could be computed using (15). However, the *nfg* decisions reported to the fusion center were not used to carry out a gateway level detection. Instead, the *nfg* decisions were used to carry out a field level decision with PD_{field} given by (17). Note that PD_g^* is not required in computing the field level probability of detection, PD_{field}.

If the target is in a certain gateway neighborhood, it is assumed that the gateway would report at least T_{fleld} detections to the fusion center. The fusion center declares the target in the gateway neighborhood that reports the greatest number of detections. Although other gateways may satisfy the threshold, the target is localized in the neighborhood of the gateway with the highest detection count; therefore, the term 'winning' gateway (the first term in (17)) may be employed. Each of the remaining gateways can report up to a maximum of 1 detection less than the number of detections reported by the winning gateway. Therefore, the second term in (17) is raised to a power of (*ng*-1) indicating that all gateways but the winning gateway are participating in the event. The most important assumption for calculation of PD_{field} using (17) is that a field level detection implies that the target is present in the 'correct' gateway neighborhood, and is absent in all the others. It is assumed that the target appears in a gateway neighborhood, and is stationary for the entire duration of the field-level epoch.

A field level false alarm occurs when the target is absent, but one or more gateways report T_{field} or more detections out of the *nfg* decisions sent to the fusion center. The probability of a gateway falsely detecting a target is PF_g*, since the target is absent. The only condition for a false alarm to occur at the field level is that one or more gateways must satisfy T_{field}, the field threshold. A discussion of false alarms at the field level follows.

IV. Specifiable Field-Level Probability of False Alarm

The false alarm performance of the sensor network (PF) at the field level can be specified, and the design parameters can be derived in a top down manner to satisfy the same [3]. This section presents an approach that may be used to achieve a specified PF at the field level.

False Alarms at the Field Level

A false alarm at the field level occurs when the fusion center declares that there is a target when there is none. Let F be the event that a false alarm occurs at the fusion center. Then let the event $UCF = (F \cap H_0)$ denote the event of an unconditional false alarm. By unconditional, we mean that it is the probability that the fusion center wrongly decides that the target is present. From Bayes' Rule, unconditional probability can be calculated as

$$Pr\{A \cap B\} = Pr\{A \mid B\} \cdot Pr\{B\}. \tag{21}$$

In terms of our problem,

$$Pr\{F \cap H_0\} = Pr\{F \mid H_0\} \cdot Pr\{H_0\}. \tag{22}$$

The probability of the event that the fusion center has declared a target detection given that the target is absent is the conditional probability of a false alarm, $Pr\{F \mid H_0\} \cdot Pr\{H_0\}$, is the prior probability of the target being absent. The prior probability is the marginal probability, interpreted as a description of what is known about a variable in the absence of some evidence or data. It is an unconditional probability. For Bayes' rule to apply, $Pr\{H_0\} \neq 0$.

Expected Number of False Alarms (ENFA)

Expectation of a random variable is the average of all possible values of a random variable, where a value is weighted according to the probability that it will appear. The expectation is sometimes also called the average, and is also known as the expected value or mean of the random variable. The expectation $E[R]$, of a random variable, R, on sample space, S, is defined as

$$E[R] = \sum_{s \in S} R(s) \cdot Pr\{s\}. \tag{23}$$

Alternately,

$$E[R] = \sum_{r \in Range(R)} r \cdot Pr\{R = r\}. \tag{24}$$

One technicality in both of the above definitions that can cause trouble if ignored is that the order of the series is not specified. The limits of the above series are not well defined unless the series are not absolutely convergent, i.e., the sum of the absolute values of the terms converges. For absolutely convergent series, the order of summation does not matter; the series converge to the same value.

Like other averages, the expected value does not say anything about what will happen in a single trial. For example, an average person in the US owns 1.3 cars. Obviously, none has exactly that number. Therefore, we do not expect to see the expected value of a random variable in one trial. However, over a large number of values, we do expect the values to average out close to the expected value.

Expected Value of an Indicator Random Variable – False Alarm

For simplicity, we assume that the target is present or absent in each decision epoch independent of the other epochs. Let I_{F_i} be 1 if the fusion center declares the target present in epoch *i*, and 0 otherwise.

The expected value of an indicator random variable is the probability of that event. Let I_F be the indicator random variable for the event of a false alarm. Thus, $I_F = 1$ if and only if a false alarm occurs, and is 0 otherwise.

$$E[\,I_F\,] = Pr\{\,UCF\,\} \tag{25}$$

Proof:

$$\begin{aligned} E[I_F] &= 1 \cdot Pr\{I_F = 1\} + 0 \cdot Pr\{I_F = 0\} \\ &= Pr\{I_F = 1\} \\ &= Pr\{UCF\} \end{aligned}$$

The number of false alarms per week is obtained by summing the false alarms over all of the W time epochs

$$I_F = \sum_{i=1}^{W} I_{F_i} \tag{26}$$

where W is the number of epochs, or decisions per week. Then the number of false alarms can be expressed as

$$\begin{aligned} E[I_F] &= E\left[\sum_{i=1}^{W} I_{F_i}\right] \\ &= \sum_{i=1}^{W} E\left[I_{F_i}\right]. \end{aligned}$$

Using (24), the expected number of false alarms per week is

$$ENFA = E[I_F] = W \cdot Pr\{UCF\}. \tag{27}$$

Using (22),

$$ENFA = W \cdot (PF_{field} \cdot (0.9))$$

$$\therefore PF_{field} = \frac{ENFA}{(0.9) \cdot W} \tag{28}$$

Here, the prior probability of the target being absent is assumed to be 0.9 [3].

False alarms follow a binomial distribution [1]. The mean and standard deviation for the binomial distribution are calculated as [4]

$$\text{mean for a binomial distribution} = np$$
$$\text{standard deviation for a binomial distribution} = \sqrt{npq}$$

where n = number of trials, p = probability of success in one trial, and q = 1-p. For an experiment, the number of observations of a successful event may be divided by the total number of observations to compute a proportion. For a binomially distributed proportion, confidence intervals (CI) can be computed as described in [5], [6].

V. Methods

The approach in [1] described a two-tier sensor network (or gateway neighborhood) to monitor an area using a large number of sensors randomly deployed over the region, and gave probabilistic measures of performance in terms of detection and false alarm probabilities. Since these performance measures apply for a gateway neighborhood, the work provided the necessary building block to test the usefulness of repeated trials in hierarchical sensor networks that have multiple gateways in an architecture having more than two tiers. This section describes the methods used in the set up of a hierarchical sensor network using multiple non-overlapping gateway areas to detect a randomly appearing target, and describes the procedures used to demonstrate the use of repeated trials to attain the degrees of freedom needed to achieve a desired level of performance using the hierarchical network.

For a three-tier network, simulations were used to demonstrate the process of selecting thresholds and numbers of trials for repeated trials at different levels and to demonstrate the validity of theoretical predictions for PD and PF at the gateway and field levels. Two levels of repeated trials (Gateway, Improved Gateway) were performed at the gateway nodes and one level of repeated trials (Field) was performed at the fusion center.

Gateway Area Setup

For the simulations, a field of 10,000 randomly placed sensors was organized into ten gateways using the following parameters.

Number of sensors per gateway neighborhood:	$N = 1000$
Number of gateway neighborhoods:	$ng = 10$
Sensor parameters:	$\alpha = 200$, $a = 100$, $n=2$, $\tau = 0.1$, and $P_0 = 1000$
Gateway scan threshold:	$TG = 473$

The gateway scan threshold (TG) was chosen from ROC data to get an exemplary level of performance (P_f and P_d).

For $\tau = 0.1$, (4) was used to calculate pfa =0.4602 for the sensors. For TG = 473, (8) and (11) were used to calculate P_f = 0.216970 and P_d = 0.821320 for the gateways. Simulations (10,000 runs) were performed to validate the theoretical gateway performance results. The gateway detection performance was simulated by making targets appear within the gateway and observing a gateway-level detection if the number of sensors reporting a detection exceeded TG. The gateway false alarm performance was simulated for no target in the gateway by observing a gateway-level false alarm if the number of sensors reporting a detection exceeded TG. The numbers of observed detections and false alarms were divided by the total number of simulation runs to compute a proportion that was compared to the theoretical probability.

Desired Gateway and Field Performances

The gateway performance P_f = 0.216970 and P_d = 0.821320 achieved using a single scan of the sensors in the gateway neighborhood was not adequate. Table 1 lists the performance specifications chosen for this demonstration of repeated trials at the gateway and field levels.

The specified field performance reflects a high detection probability ($PD_{field} > 0.95$) and a low false alarm probability ($PF_{field} < 0.0005$) that would allow an average of one false alarm in two weeks if 1008 decisions were made per week, which corresponds to one decision every 10 minutes ($ENFA$=0.5, W=1008 in (28)). The following sections describe the computation of probabilities that must be analyzed to select the thresholds and number of repeated trials needed to attain the desired performance described in Table 1.

Table 1. Desired false alarm and detection probabilities for the gateway and field

Level	PF	PD
Gateway (after a first level of repeated trials)	$PF_g < 0.025$	$PD_g > 0.9$
Improved Gateway (after a second level of repeated trials)	$PF_g^* < 0.00005$	$PD_g^* > 0.95$
Field	$PF_{field} < 0.0005$	$PD_{field} > 0.95$

Repeated Trials at the Gateway Level

First Gateway Level of Repeated Trials. Simulations were used to demonstrate the process of selecting the threshold and number of trials for the first level of repeated trials done at the gateway. For $P_f = 0.216970$ and $P_d = 0.821320$, the probabilities (PD_g and PF_g) of getting exactly a) k detections and false alarms out of n trials and b) k or more detections and false alarms out of n trials were computed using (15) and (16) for different numbers of trials. These probabilities were examined to allow the threshold, T_g, and number of repeated trials, n, to be chosen to meet the specified $PD_g > 0.9$ and $PF_g < 0.025$ (Table 1). If more than one choice for the threshold and number of repeated trials would allow the specified PD_g and PF_g to be met, the lowest number of repeated trials was selected to reduce computational and communications overhead. Simulations (10,000 runs) were performed to validate the theoretical gateway performance results that would be achieved for the selected threshold and number of repeated trials. The numbers of observed detections and false alarms were divided by the total number of simulation runs to compute a proportion that was compared to the theoretical probability.

Second Gateway Level of Repeated Trials. Simulations were also used to demonstrate the process of selecting the threshold and number of trials for a second level of repeated trials done at the gateway level. Using the performances attained in the first level of repeated trials (the PD_g and PF_g achieved using the selected threshold, T_g, and number of repeated trials, n) the probabilities (PD_g^* and PF_g^*) of getting exactly a) k detections and false alarms out of n trials and b) k or more detections and false alarms out of n trials were computed using (15) and (16) (or (18)) for different numbers of trials. These probabilities were examined to allow a threshold, T_g^*, and number of repeated trials, n^*, to be chosen to meet the specified $PD_g^* > 0.95$ and $PF_g^* < 0.00005$ (Table 1). If more than one choice for the threshold and number of repeated trials would allow the specified performances to be met, the lowest number of repeated trials was selected to reduce computational and communications overhead. Simulations (10,000 runs) were performed to validate the theoretical gateway performance results that would be achieved for the selected threshold and number of repeated trials. The numbers of observed detections and false alarms were divided by the total number of simulation runs to compute a proportion that was compared to the theoretical probability.

The Field Level

Gateway level detection decisions made using a threshold T_g in one level of n repeated trials were fused to make field level detection decisions. Each of the ng gateways sent multiple (nfg) decisions to the field level fusion center over a single time epoch. The nfg decisions were used to make the field level detection decisions. Equations (17), (18), and (20) were used to compute PD_{field} and PF_{field} using the probabilities of detection and false alarm for the identical gateways, PD_g and PF_g, respectively, achieved using the threshold, T_g, and number of repeated trials, n, through the first level of repeated trials at the gateway level. The threshold, T_{field}, was chosen to allow the expected number of false alarms ($ENFA$) specification to be met. The theoretical field performance (PD_{field} and PF_{field}) was validated using detailed simulations. At each sensor, a noisy signal, s_i, was generated and compared to the threshold, τ, to make a detection decision. Repeated trials were simulated at all the gateways in the field allow the observed field performance to be calculated. A detection decision was made for a gateway if the gateway not only met the threshold, T_{field}, but also had the greatest number of detections among all the gateways.

VI. Results

Gateway Setup and Performance for a Single Trial

The gateway ROC curve shows the operating point (TG = 473, P_f = 0.2169, P_d = .8213) when a single scan was used by the gateway to make a decision about the presence of the target (Fig. 3). The P_d and P_f values calculated using (11) and (8) were within the 95% confidence intervals for the proportions for detections and false alarms observed in simulations (Table 2). Thus, the simulation results supported the theory.

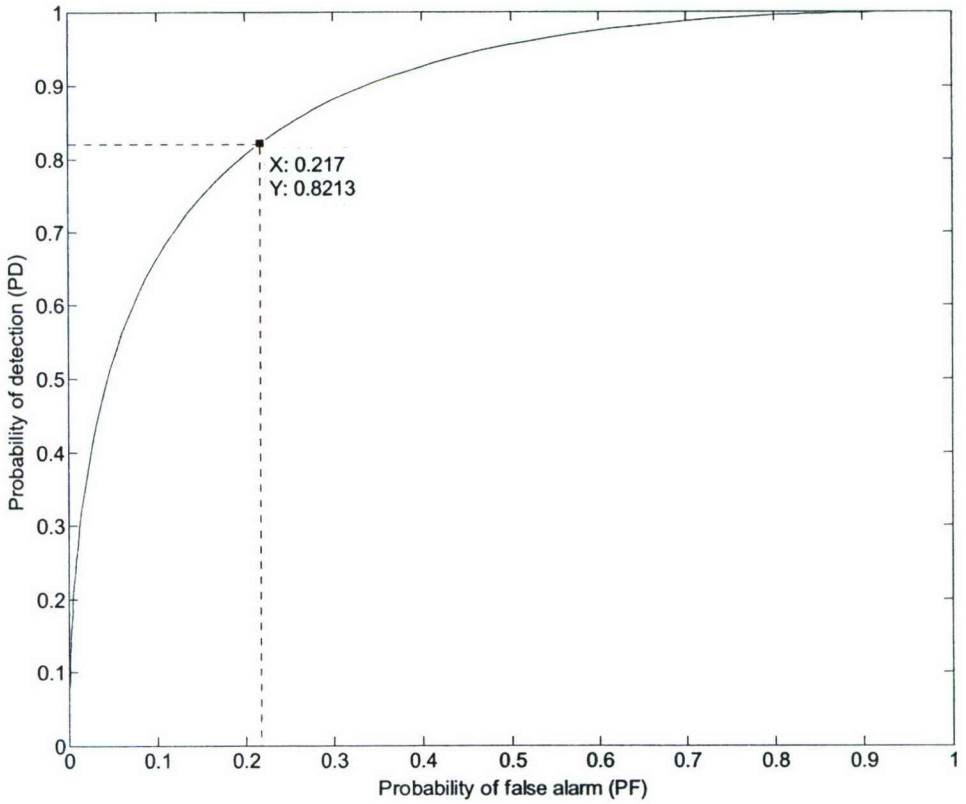


Fig. 3. Receiver Operating Characteristic (ROC) for the gateway, and the selected operating point.

Table 2. Gateway level performance for a single scan, TG = 473, N = 1000

Event	Proportion in 10,000 Simulations	95% C.I.		Theoretical Probability
		Lower Bound	Upper Bound	
Detection	0.8215	0.8139	0.8289	P_d = 0.8213
False alarm	0.2189	0.2109	0.2271	P_f = 0.2169

Repeated Trials at the Gateway Level

First Gateway Level of Repeated Trials. A single gateway trial achieved P_d = 0.821320 and P_f = 0.216970. Table 3 shows the theoretical probabilities for a first level of repeated trials performed to demonstrate improved performance at the gateway for 2, 6, and 10 trials. Tables

3(a) and 3(b) show the probabilities of getting exactly k false alarms and detections in n trials, respectively. Tables 3(c) and 3(d) show the probabilities of getting k or more detections and false alarms out of n trials. Thus, Tables 3(c) and 3(d) show the resulting probabilities (PF_g and PD_g) attained for each choice of k and n, if the target is declared present only after receiving k or more reports of the target being present. These tables show that the performance is enhanced by the voting carried out through repeated trials. A review of the probabilities determined that a choice of $T_g = 4$ for the threshold for $n = 6$ repeated trials attained $PD_g > 0.9$ and $PF_g < 0.025$ as specified (Table 1). Thus, a target is declared present only after receiving $T_g = 4$ or more reports of the target being present out of the $n = 6$ trials. The highlighted and underlined values in Table 3 show the resulting gateway performance, $PD_g = 0.9255$ and $PF_g = 0.0227$, achieved using $T_g = 4$ in 6 repeated trials. A higher PD_g and lower PF_g could be achieved by choosing 10 trials and a threshold of 6. However, since more trials take longer to complete, involve more data fusion operations, and require more communications, 4 was chosen as the threshold in 6 repeated trials.

Table 3. Theoretical performance at the gateway level through first level of repeated trials

Exact Probabilities (k out of n) of False Alarm, $P_f = 0.216970$				Exact Probabilities (k out of n) of Detection, $P_d = 0.821320$		
k	n=2	n=6	N=10	n=2	n=6	n=10
1	0.339788	0.383215	0.240107	0.293507	0.000898	0.000002
2	0.047076	0.265463	0.299391	0.674567	0.010314	0.000032
3		0.098076	0.221222		0.063211	0.000387
4		0.020382	0.107273		0.217918	0.003110
5		0.002259	0.035669		0.400672	0.017153
6		0.000104	0.008236		0.306955	0.065705
7			0.001304			0.172582
8			0.000136			0.297484
9			0.000008			0.303870
10			0			0.139677
	(a)				(b)	
Probability of >= k False Alarms, PF_g				Probability of >= k Detections, PD_g		
k	n=2	n=6	N=10	n=2	n=6	n=10
1	0.386864	0.769500	0.913347	0.968073	0.999967	1
2	0.047076	0.386285	0.673240	0.674567	0.999070	0.999998
3		0.120822	0.373848		0.988756	0.999967
4		**0.022745**	0.152626		**0.925545**	0.999580
5		0.002363	0.045353		0.707627	0.996471
6		0.000104	0.009684		0.306955	0.979317
7			0.001448			0.913613
8			0.000144			0.741030
9			0.000009			0.443546
10			0			0.139677
	(c)				(d)	

For 6 repeated trials, and thresholds of 4, 3, and 2, the values of PD_g and PF_g calculated using (15) and (16) were within the 95% confidence intervals for the proportions for detections and false alarms observed in simulations (Tables 4-7). Thus, these simulation results demonstrated the validity of the theoretical detection probabilities. The only exception was for a threshold of 1, where $PD_g = 0.99997$ is above 0.9998, the upper bound for the 95% confidence interval (Table 7). Several facts suggest that there is no reason to question the agreement of the results with theory. First, out of about 20 comparisons to a 95% CI made in this report, only PD_g

= 0.99997 is outside the CI; for a 95% CI, about one of twenty comparisons would be expected to be outside the CI. In addition, the CI was computed for the proportions observed in 10,000 trials. An alternative approach was to compute the CI based on the proportion expected for the theoretical probability; this alternative was not used for this report since a) the theoretical probabilities were often so close to 1 or so close to zero that computing the CI based on the expected proportions out of 10,000 trials would have required careful rounding, and b) the two alternative methods for computing the CI were expected to yield similar results. However, it is useful to apply the alternative approach for computing a CI for PD_g = 0.99997. The proportion (10,000 observations out of 10,000 trials) corresponding to PD_g = 0.99997 is 1. For a proportion of 1, the 95% CI (0.9995 to 1) contains the value 0.9996 observed in 10,000 simulation trials. Thus, the two alternative methods for computing the CI gave different results. Even though there is no reason to question the agreement of the results with theory, additional simulations could be performed to give a more compete test for the theory.

Table 4. Gateway level performance for the first level of repeated trials.
(n=6 scans for a gateway, threshold (T_g) = 4 out of 6 trials)

Event	Proportion in 10,000 Simulations	95% C.I.		Theoretical Probability
		Lower Bound	Upper Bound	
Detection	0.9254	0.9201	0.9304	PD_g = 0.9255
False alarm	0.0223	0.0196	0.0254	PF_g = 0.02270

Table 5. Gateway level performance after repeated trials.
(n=6 scans for a gateway, threshold (T_g) = 3 out of 6 trials)

Event	Proportion in 10,000 Simulations	95% C.I.		Theoretical Probability
		Lower Bound	Upper Bound	
Detection	0.9891	0.9869	0.9910	PD_g =0.9888
False alarm	0.1234	0.1171	0.1300	PF_g = 0.1208

Table 6. Gateway level performance after repeated trials.
(n=6 scans for a gateway, threshold (T_g) = 2 out of 6 trials)

Event	Proportion in 10,000 Simulations	95% C.I.		Theoretical Probability
		Lower Bound	Upper Bound	
Detection	0.9994	0.9987	0.9997	PD_g = 0.9991
False alarm	0.3848	0.3753	0.3944	PF_g = 0.3862

Table 7. Gateway level performance after repeated trials.
(n=6 scans for a gateway, threshold (T_g) = 1 out of 6 trials)

Event	Proportion in 10,000 Simulations	95% C.I.		Theoretical Probability
		Lower Bound	Upper Bound	
Detection	0.9996	0.9990	0.9998	PD_g = 0.99997
False alarm	0.7731	0.7648	0.7812	PF_g = 0.76950

Second Gateway Level of Repeated Trials. A first level of repeated trials at the gateway using a threshold T_g = 4 detections out of n = 6 repeated trials gave PF_g = 0.022745 and PD_g = 0.925545. Table 8 shows theoretical data for a second level of repeated trials performed to demonstrate improved performance at the gateway. Probabilities of false alarm and detection were calculated using different thresholds for 2, 4, and 6 repeated trials. Tables 8(a) and (b) show

the probabilities of getting exactly k false alarms and detections in n trials. Tables 8(c) and (d) show the resulting performance (PF_g* and PD_g*) with the choice of k. Thus, PF_g* and PD_g* are the resulting probabilities if the target is declared present only after receiving k or more detections out of the n trials. A review of the probabilities determined that a choice of T_g* = 3 for the threshold for n* = 4 repeated trials attained PD_g* > 0.95 and PF_g* < 0.00005 as specified (Table 1). Thus, a target is declared present only after receiving T_g* = 3 or more reports of the target being present out of the n* = 4 trials. The highlighted and underlined values in Table 8 show the resulting gateway performance, PD_g* = 0.969948 and PF_g* = 0.000046, achieved using T_g* = 3 in 4 repeated trials. A higher PD_g* and lower PF_g* could be achieved by choosing 6 trials and a threshold of 4. However, since fewer trials are more economical, 3 was chosen as the threshold in 4 repeated trials.

Table 8. Theoretical performance at the gateway level through a second level of repeated trials

Trials	Exact Probabilities of False Alarm each trial = 6 scans PF_g = 0.022745			Exact Probabilities of Detection each trial = 6 scans PD_g = 0.925545		
k	n=2	n=4	n=6	n=2	n=4	n=6
1	0.044455	0.084912	0.121640	0.137823	0.001528	0.000013
2	0.000517	0.002964	0.007078	0.856634	0.028493	0.000395
3		0.000046	0.000220		0.236127	0.006545
4		0.000000	0.000004		0.733821	0.061020
5			0.000000			0.303412
6			0.000000			0.628616
	(a)			(b)		
Trials	Probability of >= k False Alarms, PF_g*			Probability of >= k Detections, PD_g*		
k	n=2	n=4	n=6	n=2	n=4	n=6
1	0.044973	0.087923	0.128941	0.994456	0.999969	1.000000
2	0.000517	0.003011	0.007301	0.856634	0.998441	0.999987
3		**0.000046**	0.000224		**0.969948**	0.999592
4		0.000000	0.000004		0.733821	0.993047
5			0.000000			0.932028
6			0.000000			0.628616
	(c)			(d)		

For 4 repeated trials, and thresholds of 3, 2, and 1, the values of PD_g*and PF_g* calculated using (15) and (16) (or (18)) were within the 95% confidence intervals for the proportions for detections and false alarms observed in simulations (Tables 9-11). Thus, the simulation results demonstrated the validity of the theoretical detection probabilities.

Table 9. Gateway level performance validation
(n = 4 sets of 6 scans for a gateway; threshold (T_g*) = 3 out of 4 trials)

Event	Proportion in 10,000 Simulations	95% C.I.		Theoretical Probability
		Lower Bound	Upper Bound	
Detection	**0.9700**	**0.9665**	**0.9732**	**PD_g*=0.96995**
****False alarm**	**0.0000**	**0.0000**	**0.0004**	**PF_g*=0.00005**

** Note: In the table above, the theoretical false alarm probability is $4.6 \times 10^{-5} \sim 5 \times 10^{-5}$, which implies 5 false alarms in 100,000 trials. Since the number of simulations performed was 10,000, the number of trials did not allow comparison of the theoretical data with simulations.

Table 10. Gateway level performance validation
(n = 4 sets of 6 scans for a gateway; threshold (T_g*) = 2 out of 4 trials)

Event	Proportion in 10,000 Simulations	95% C.I.		Theoretical Probability
		Lower Bound	Upper Bound	
Detection	0.9985	0.9975	0.9991	PD_g*=0.9984
False alarm	0.0029	0.0020	0.0042	PF_g*=0.0030

Table 11. Gateway level performance validation
(n = 4 sets of 6 scans for a gateway; threshold (T_g*) = 1 out of 4 trials)

Event	Proportion in 10,000 Simulations	95% C.I.		Theoretical Probability
		Lower Bound	Upper Bound	
Detection	1.0000	0.9996	1.0000	PD_g*=0.9999
False alarm	0.0860	0.0807	0.0917	PF_g*=0.0879

Field Level Performance

Decisions made in a first level of repeated trials at the gateway using a threshold $T_g = 4$ detections out of $n = 6$ repeated trials gave $PF_g = 0.022745$ and $PD_g = 0.925545$. These gateway level decisions were fused to make field level detection decisions. Tables 12 and 13 compare the theoretical field level detection performance with the results of simulations for thresholds $T_{field} = 2$ and $T_{field} = 3$ out of $nfg = 4$ trials. A choice of $k = T_{field} = 3$ gave the underlined and bolded $PF_{field} = 0.0004$ and $PD_{field} = 0.9699$ which met the specified $PD_{field} > 0.95$ and $PF_{field} < 0.0005$ (Table 1). The target was declared present only after receiving $T_{field} = 3$ or more reports of the target being present out of the $nfg = 4$ trials. Thus, the simulation results demonstrated the validity of the theoretical detection probabilities.

Table 12. Field level performance
(n = 4 sets of 6 scans for a gateway; threshold (T_{field}) = 2 out of 4 trials)

Event	Proportion in 10,000 Simulations	95% C.I.		Theoretical Probability
		Lower Bound	Upper Bound	
Detection	0.9975	0.9963	0.9984	PD_{field} =0.9983
False alarm	0.0281	0.0250	0.0316	PF_{field} =0.0297

Table 13. Field level performance
(n = 4 sets of 6 scans for a gateway; threshold (T_{field}) = 3 out of 4 trials)

Event	Proportion in 10,000 Simulations	95% C.I.		Theoretical Probability
		Lower Bound	Upper Bound	
Detection	**0.9723**	**0.9689**	**0.9753**	***PD*$_{field}$ =0.9699**
False alarm	**0.0004**	**0.0002**	**0.0010**	***PF*$_{field}$ =0.0004**

VII. Discussion

Repeated trials improved detection performance in a hierarchical network, and provided the degrees of freedom needed to attain a specified field performance using common sensors having a relatively poor performance. With repeated trials, an impressive field performance of PF_{field} = 0.0004 (about one false alarm in two weeks) and PD_{field} = 0.9699 was achieved using low quality common sensors having *pfa* =0.4602 and gateways for which P_f = 0.216970 and P_d = 0.821320 without repeated trials. More repeated trials could achieve an even better field performance using the same common sensors.

However, the use of repeated trials requires additional detection decisions at the appropriate hierarchical levels, and communication of the extra decisions. Additional decisions and associated communications introduce delays. Future work should consider the performance tradeoffs involved in choosing the common sensor performance, the number of repeated trials, and the hierarchical level at which trials are repeated.

In addition, the routine use of a sensor network involves finite numbers of observations in which the frequency of an event is a random variable having the theoretical event probability as the expected value. For a theoretical single-trial gateway performance P_d = 0.8213 and P_f = 0.2169, the achieved PF_{field} = 0.0004 met the specified criterion $PF_{field} < 0.0005$, as desired. However, the upper bound for the 95% confidence interval for PF_{field} was 0.001 (Table 13), a number that was twice the value of the specified PF_{field}. Although the specified performance was achieved in this work for a simulation trial of 10,000 runs, corresponding to a period of about ten weeks, the performance attained in additional simulation trials would sometimes be better and sometimes be worse. Due to the inherent nature of the detection process, the performance predicted is the average performance that can be expected from the system, therefore implying that the short-term performance can sometimes be worse and sometimes be better.

The performance of the lower and middle tiers can also affect the overall performance of a multi-tier system. For example, (17) and (20) show that PD_{field} and PF_{field} depend on P_d and P_f for a single trial at the gateway level. If the actual gateway performance differs from the theoretical probabilities P_d = 0.8213 and P_f = 0.2169, the PD_{field} and PF_{field} would differ from the theoretical values.

Future work should consider the confidence intervals for the probabilities at all hierarchical levels in outlining a conservative design methodology that guarantees a worst-case field performance that meets the specification over an appropriate period of use. The approach presented in this report could be used as a starting point; the design would be carried out to achieve a performance that exceeds the specification such that the worst-case performance achieved would meet the specification. In addition, further work could include the development of test cases for rigorous characterization of the system performance and validation of the theory.

VIII. Conclusions

The work in [1] described detection performance measures for two-tier sensor networks. However, that approach does not provide the degrees of freedom necessary to obtain a desired level of performance, and does not consider hierarchical sensor networks that have more than

two layers. This report extended the work in [1] and [3] by making use of repeated trials to provide the degrees of freedom needed to give improved performance using a distributed multi-tier hierarchical sensor network. Probabilistic performance measures were developed and applied for repeated trials in multi-tier hierarchical sensor networks. Repeated trials achieved an impressive field performance in a hierarchical network using a large number of low quality sensors.

IX. References

[1] D. Klamer, M. Moore, P. K. Varshney, and R. Niu, "Decision fusion in a wireless sensor network with a large number of sensors," Proceedings of the Seventh International Conference on Information Fusion, Stockholm, Sweden, June 2004.

[2] T. Clouqueur, P. Ramanathan, K. K. Saluja, and Kuang-Ching Wang, "Value-fusion versus decision-fusion for fault-tolerance in collaborative target detection in sensor networks," Proceedings of the Fourth International Conference on Information Fusion, Montreal, Canada, August 2001.

[3] D. Klamer, M. Moore, P. K. Varshney, and R. Niu, "Sensor systems with a large number of autonomous distributed sensors," ONR Presentation, Spring 2004.

[4] A. Papoulis, *Probability, Random Variables, and Stochastic Processes*, Third edition, McGraw-Hill, Inc., New York City, New York, 1991.

[5] R. G. Newcombe, "Two-Sided Confidence Intervals for the Single Proportion: Comparison of Seven Methods," *Statistics in Medicine,* 17, 857-872, 1998.

[6] E. B. Wilson, "Probable inference, the law of succession, and statistical inference," *Journal of the American Statistical Association* 22: 209-212, 1927.

A Simple Two-Step Approach for Maximum Likelihood Target Localization Using Binary Data

Parag P. Joshi
Department of Electrical and Computer Engineering,
The University of Alabama at Birmingham,
Birmingham, AL, USA.

Thomas C. Jannett
Department of Electrical and Computer Engineering,
The University of Alabama at Birmingham,
Birmingham, AL, USA.

Abstract - *Localizing a target using a maximum likelihood framework involves optimization of a complex multimodal function. Deterministic search algorithms are ineffective because they can converge to local minima, resulting in large errors. Stochastic algorithms provide global solutions with higher accuracy, but can be computationally burdensome, and can require tuning of parameters. This paper presents a novel two-step technique that allows accurate localization of the target without a high computational burden. First, a weighted average of the positions of detecting sensors forms a coarse estimate. The coarse estimate is used as the initial estimate for a Nelder-Mead direct search. This two-step process reduces the global search to a local search, avoiding pitfalls due to convergence to local minima. For an exemplary sensor network, the deterministic direct search often fails by converging to a local minimum if the starting point is chosen at random. In contrast, the two-step algorithm accurately localizes the target with a low computational overhead.*

Keywords: Localization, maximum likelihood estimation, sensor networks, weighted average.

1 Introduction

Recent advances in sensing and wireless communications technology along with an ever expanding set of potential application areas continue to fuel the flurry of interest in wireless sensor networks (WSN) [1]. A WSN is a collection of spatially distributed sensing devices that function cooperatively to garner information about an environment or event of interest. Their ability to autonomously perform tasks such as data gathering, communication, co-ordination, and control makes WSNs a prudent deployment alternative for environments with limited accessibility.

Applications of sensor networks include environmental and habitat monitoring, seismic detection, emergency medical response, traffic surveillance, building and structure monitoring, target localization and tracking. Some attributes unique to WSNs are sensor failures, large scale of deployment, harsh ambient conditions, and small-scale sensor nodes. Owing to the limited battery life typically available to sensor nodes, minimizing the demand placed on resource utilization by any activity is vital. Hence, a lot of research in WSNs principally focuses on designing energy aware algorithms and protocols.

Target localization is an important application of WSNs. Various ways to tackle this problem have been proposed in the literature, including an approach using binary sensor data, for which the maximum likelihood (ML) estimator and its Cramer-Rao lower bound (CRLB) were derived [2]. In this approach, each sensor makes a decision about a target's presence by comparing the intensity of a measured signal to a threshold. The sensor communicates a one-bit decision to the fusion center. On receiving such binary information from all the sensors, and using *a priori* information about the positions of the sensors, the fusion center localizes the target through the non-linear optimization of a highly complex multimodal function. The approach has recently been extended for localization based on quantized data [3]. These methods are attractive because they offer the promise of accurate target localization and require that only quantized data be transmitted, which results in savings in communication bandwidth. However, optimization of a complex multimodal function is challenging. Deterministic search algorithms are ineffective because they can converge to local minima, resulting in large localization errors. Stochastic algorithms provide global solutions, but can be computationally burdensome, and can require tuning of parameters.

Some examples of deterministic optimization algorithms are gradient-based algorithms that rely on the knowledge of the first partial derivatives of the objective function, and quasi-Newton Raphson (QNR) strategies that operate on approximations of the second partial derivatives of the objective function. Deterministic strategies have the drawback of not being able to cope with multidimensional objective functions having multiple local minima or expansive flat surfaces, and can only converge to a minimum local to the starting point of the search. In contrast to traditional numerical techniques, stochastic global optimization algorithms like genetic algorithms (GAs) and

particle swarm optimization (PSO) algorithms can be used to avoid traps due to local minima. For example, GAs do not require the evaluation of gradients, and do not pose differentiability or continuity requirements on the objective function. GAs iteratively refine a single solution vector as they search for minima, and operate on entire populations of candidate solutions in parallel. This parallel nature is the inherent forte of GAs that makes them much less likely to get trapped in local minima, and also less sensitive to initial conditions. Nevertheless, GAs suffer from slow convergence rates because a lot of time is expended in testing the fitness of suboptimal solutions. Moreover, GAs, due to their stochastic nature, can only estimate the global minimum whereas traditional deterministic algorithms can find it exactly. GAs and other stochastic optimization schemes suffer from slow convergence rates, high computational costs and poor accuracy of final solution since they perform a random search of the solution space. A hybrid optimization scheme that combines both stochastic and deterministic schemes to achieve high convergence rates and avoid local traps has been shown to achieve high estimation accuracy at a high computational cost [4].

In contrast to stochastic algorithms, a deterministic algorithm used in concert with a good initial guess close to the global minimum is expected to yield a significantly higher accuracy of the final solution with less computation. This paper presents a simple, yet effective two-step approach that can be used to localize a target more accurately and with a low computational burden. The steps involved in this approach are

1) Form an initial coarse estimate of the target position using a heuristic weighted averaging (WA) technique,
2) Use the coarse estimate as the initial estimate for a deterministic Nelder-Mead search to find the global minimum.

This paper compares the results of the two-step approach with those obtained with WA alone and with a one-step deterministic Nelder-Mead search using a random guess as the initial value. The paper is organized as follows. Section 2 describes the ML target localization framework. The WA method is outlined in section 3. The methods used to compare the localization performance of one and two-step approaches and WA are described in section 4. The results and discussion are presented in section 5 followed by the conclusion in section 6.

2 Maximum likelihood target localization

The maximum likelihood (ML) estimation framework for target localization using binary data presented in [2] is utilized in this paper and reviewed in this section.

2.1 Sensor detection model

The sensor detection model assumes that signal intensity attenuates as the distance from the target to a sensor increases

$$a_i = \sqrt{\frac{P_o}{1+\alpha d_i^n}} \tag{1}$$

where d_i is the distance from the target to the ith sensor, a_i is the signal amplitude at the ith sensor, P_o is the signal power received at the ith sensor, α is a constant, and n is the decay exponent. Values of parameters used in this paper are $n = 2$, $\alpha = 2$, and $P_o = 64$. The distance of the target from the ith sensor is given by

$$d_i = \sqrt{(x_i - x_t)^2 + (y_i - y_t)^2} \tag{2}$$

where (x_i, y_i) are the coordinates of the ith sensor and the target location is given by (x_t, y_t). The signal a_i is corrupted by standard Gaussian noise ω_{ij} that is independent across sensor i and time frame j

$$s_{ij} = a_i + \omega_{ij}. \tag{3}$$

For a given timeframe j, sensor i makes a binary decision about the presence of the target by comparing the noise corrupted signal s_{ij} to a common preset threshold $\eta_{ij} = \eta_j$

$$\eta_j = \frac{j}{T+1}\sqrt{P_o}. \tag{4}$$

The threshold has a significant effect on the coarse estimate of the target position. If the threshold is low, a high number of sensors detect the target and the accuracy of the estimate is poor. On the other hand, for a high threshold, there may be no detections at all. Allowing the threshold to vary with the timeframe as in (4) provides a wide range of thresholds, making it likely that the target position can be estimated accurately.

After collecting the decisions I_{ij} from all N sensors for all T timeframes, the higher-level node can estimate the parameter vector including the target position $\theta = [x_t\ y_t]$ by maximizing the log-likelihood function (5) with respect to θ (Fig. 1).

$$\ln p(I \mid \theta) = \sum_{i=1}^{N}\sum_{j=1}^{T} I_{ij} \ln\left[Q\left(\eta_{ij} - a_i(\theta)\right)\right] + \left(1 - I_{ij}\right)\ln\left[\left(1 - Q\left(\eta_{ij} - a_i(\theta)\right)\right)\right] \tag{5}$$

The $Q(\cdot)$ function is the complementary distribution of the standard Gaussian distribution function defined by

$$Q(x) = \int_x^{\infty} \frac{1}{\sqrt{2\pi}} e^{\frac{-t^2}{2}}\, dt. \tag{6}$$

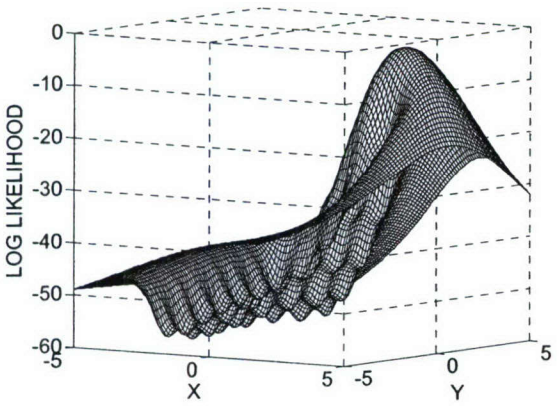


Fig 1. Plot of the log-likelihood function (target placed at x_t=y_t=2.5, N=49, P_o=64, n=2, α=2).

The ML estimate $\hat{\theta}$ corresponds to the global maximum of the log-likelihood function (5)

$$\hat{\theta} = \max_{\theta} \ln p(I \mid \theta). \tag{7}$$

2.2 The Cramer-Rao lower bound

Since the ML estimator is an asymptotically unbiased estimator, the Cramer Rao Lower Bound (CRLB) for an unbiased ML estimate is given by

$$E\left\{\left[\hat{\theta}(I)-\theta\right]\left[\hat{\theta}(I)-\theta\right]^T\right\} \geq J^{-1} \tag{8}$$

where J is the Fisher Information Matrix. Details of the Fisher Information Matrix are given in [2]. The covariances of the errors in the target's x and y position estimates are bounded by the (1,1) and (2,2) elements of the J^{-1} matrix, respectively.

$$\begin{aligned} \text{var}(\hat{\theta}_1) &= \text{var}(\hat{x}_t) \geq J_{11}^{-1} \\ \text{var}(\hat{\theta}_2) &= \text{var}(\hat{y}_t) \geq J_{22}^{-1} \end{aligned} \tag{9}$$

2.3 Network Layout

Consider a network having N=49 sensors as shown in Fig. 2. The sensors are uniformly placed on a grid of unit width.

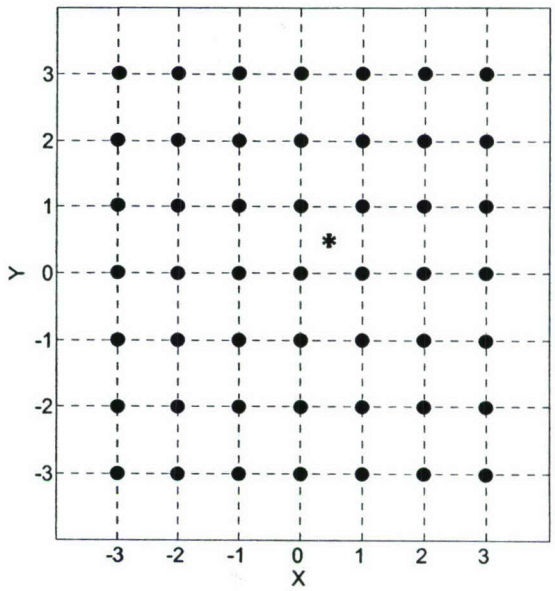


Figure 2. Layout of the sensor network with circles showing sensor positions. A target at (0.5, 0.5) is shown as an asterisk.

3 Weighted average estimation

Using the WA technique, the first step is to compute the mean of the positions of all the sensors detecting the target in each timeframe. Let the estimated target position in timeframe j be denoted as $(\bar{x}_j, \bar{y}_j)$. An overall estimate of the target position, denoted by $(\hat{x}_t, \hat{y}_t)$, is then computed by performing a weighted average (WA) of the estimates from all the timeframes

$$\hat{x}_t = \frac{\sum_{j=1}^{T} w_j \cdot \bar{x}_j}{\sum_{j=1}^{T} w_j}; \quad \hat{y}_t = \frac{\sum_{j=1}^{T} w_j \cdot \bar{y}_j}{\sum_{j=1}^{T} w_j} \tag{10}$$

where w_j is the weight for timeframe j. The local sensor threshold η_j is varied as in (4). At a low threshold, a sensor can detect a target farther away than at a higher threshold, but the probability of false alarm is higher in the earlier case. Since the number of sensors reporting the same target is greater at a lower threshold than at a higher threshold, the weight per time frame estimate needs to vary as a function of the threshold. For the purpose of our simulations, the weights for each timeframe are chosen to be equal to the square of the signal power level required to produce a detection.

$$w_j = \eta_j^2 \tag{11}$$

4 Comparison of the one and two-step approaches

Extensive Monte Carlo simulations were run to compare the quality of the ML estimates obtained using the one-step and two-step approaches. The field was set up as shown in Fig. 2, with 49 sensors laid on a uniform grid. In the one-step approach, the target position was estimated by minimizing the negative of the log-likelihood function, with the starting guess for θ chosen at random. The first step in the two-step approach was to compute a WA estimate of the target position, and the second step was to minimize the negative of the log-likelihood function using the WA estimate as the starting point. The algorithm used for minimizing the negative of the log-likelihood function in both the one-step and two-step approaches was a Nelder-Mead unconstrained minimization algorithm. The CRLB for the RMS error was computed using (9) for each timeframe at each target location. The noise in (3) was zero mean, and had unit variance (i.i.d. across all sensors and timeframes).

First, extensive Monte Carlo simulations were run for five through ten timeframes for a target position of $x_t = y_t = 0.5$, in order to facilitate comparisons with the data in [2]. Next, Monte Carlo simulations were run for ten timeframes to compare the one and two-step approaches, where x_t and y_t in each simulation were uniformly random, and were between -3 and +3. Corresponding to the square root of the normalized estimation error squared (NEES), the normalized RMS error (nRMSE) was computed by scaling the estimation error at each target location by the CRLB for the RMS error [5]. Finally, Monte Carlo simulations were run with for ten timeframes to allow the two approaches to be compared for another target position, $x_t = y_t = -1.5$.

5 Results and discussion

Tables I through III show data for representative sets of simulations. For the target position of $x_t = y_t = 0.5$, the one-step approach often failed by converging to a local minimum (Table I).

TABLE I
RELATIVE PERFORMANCE OF THE ONE AND TWO-STEP APPROACHES IN 100 MONTE CARLO SIMULATIONS FOR A TARGET POSITION (0.5, 0.5)

Total Number of Timeframes (T)	Number of Failures of One-Step Approach	Number of Failures of Two-Step Approach
5	11	0
6	2	0
7	8	0
8	7	0
9	11	0
10	7	0

With the two-step approach, RMS errors in the estimated target position were less than those achieved with WA used alone (Figs. 3 and 4), and were close to the CRLB. The agreement of these results with the results in [2] validates the methods used in the present paper. Since the one-step approach often converged to local minima, RMS errors in the estimated target position were too high to be shown on the same scale as the results of WA and the two-step approach in Figs. 3 and 4.

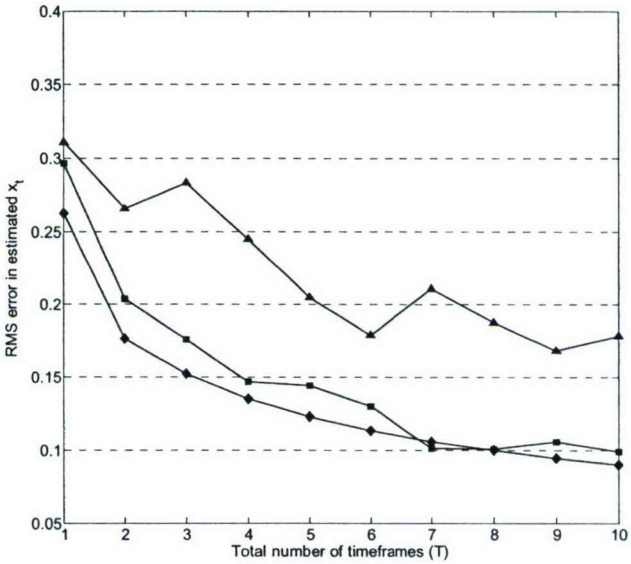


Fig. 3. Root mean square (RMS) errors in estimated x_t in 100 Monte Carlo runs for WA (triangles) and the two-step approach (squares) along with the CRLB (diamonds). Sensor network layout is shown in Fig. 2. (x_t=y_t=0.5, N=49, P_o=64, n=2, α=2).

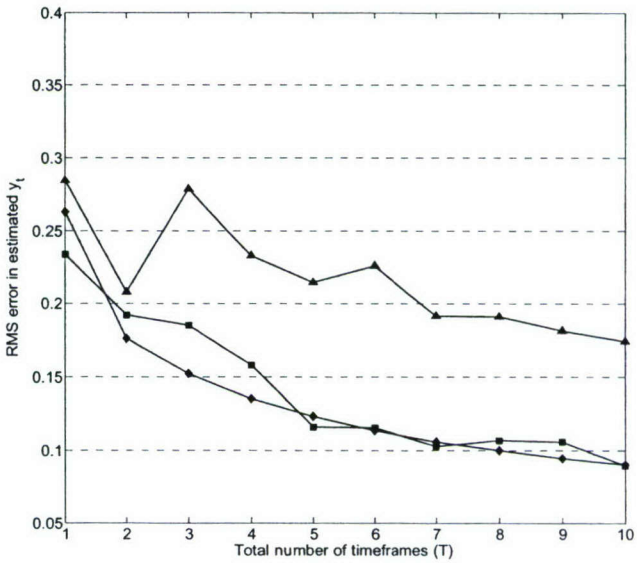


Fig. 4. Root mean square (RMS) errors in the estimated y_t in 100 Monte Carlo runs for WA (triangles) and the two-step approach (squares) along with the CRLB (diamonds). Sensor network layout is shown in Fig. 2. (x_t=y_t=0.5, N=49, P_o=64, n=2, α=2).

With the target position varied randomly throughout the field (x_t and y_t were uniformly distributed between -3 and +3), the one-step approach often converged to local minima (Table II), resulting in large RMS errors in the estimated target position.

TABLE II
RELATIVE PERFORMANCE OF THE ONE AND TWO-STEP APPROACHES IN 500 MONTE CARLO SIMULATIONS FOR RANDOM TARGET POSITIONS WITHIN THE FIELD

Total Number of Timeframes (T)	Number of Failures of One-Step Approach	Number of Failures of Two-Step Approach
10	56	0

The average nRMSE for the two-step approach over 500 runs was near unity, indicating that the RMS error in the position estimates was near the CRLB for the target positions. Convergence of the deterministic algorithm used in the second stage of the two-step approach was achieved within an average of about 80 function evaluations. The two-step approach performed better than WA (Fig. 5).

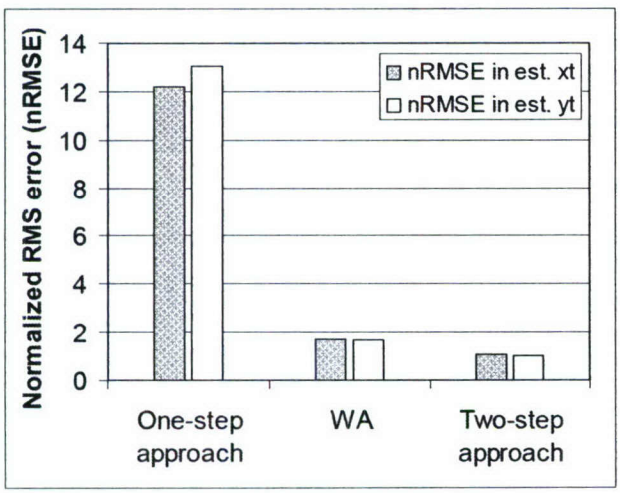


Fig. 5. Mean of the nRMSE for the one-step, WA and two-step approaches over 500 runs. Target position coordinates x_t and y_t were randomly varied between -3 and +3 according to a uniform distribution. (N=49, P_o=64, n=2, α=2, T=10).

Fig. 6 shows the results of location estimation using the one-step approach with the target placed at x_t=y_t=-1.5. The ellipse represents a 99% confidence region of the position estimates based on the CRLB. About 15 of the 100 estimated positions lie outside of the 99% confidence interval. The one-step approach gave good performance when it did not converge to local minima.

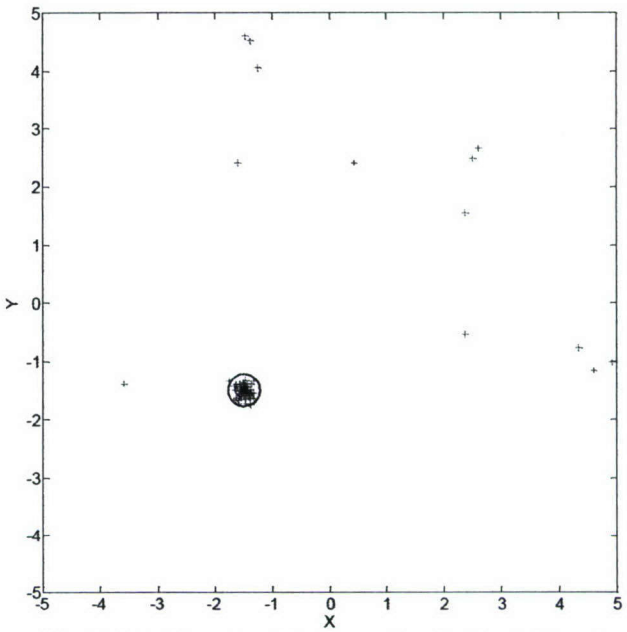


Fig. 6. Actual target position (square) and estimated target positions (plus signs) using the one-step approach for 100 Monte Carlo runs. (x_t = y_t = -1.5, N=49, P_o=64, n=2, α=2, T=10). The ellipse around the actual target position shows a 99% confidence region based on the CRLB.

The WA scheme (Fig. 7) gave estimates near the target location but its performance did not approach the CRLB.

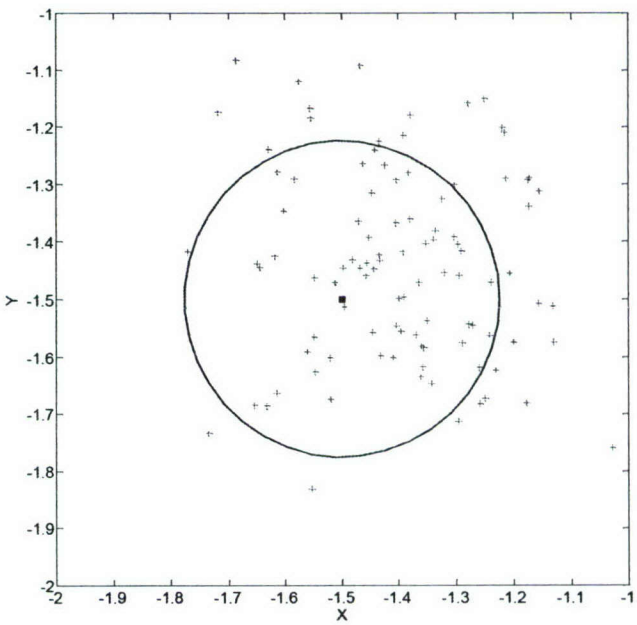


Fig. 7. Actual target position (square) and estimated target positions (plus signs) using the WA approach for 100 Monte Carlo runs. (x_t=y_t =-1.5, N=49, P_o=64, n=2, α=2, T=10). The ellipse shows a theoretical 99% confidence region based on the CRLB.

Using the WA estimate as a starting point for the Nelder-Mead search, the two-step approach performed well, with 99 of 100 estimated positions falling within the 99% confidence interval (Fig. 8).

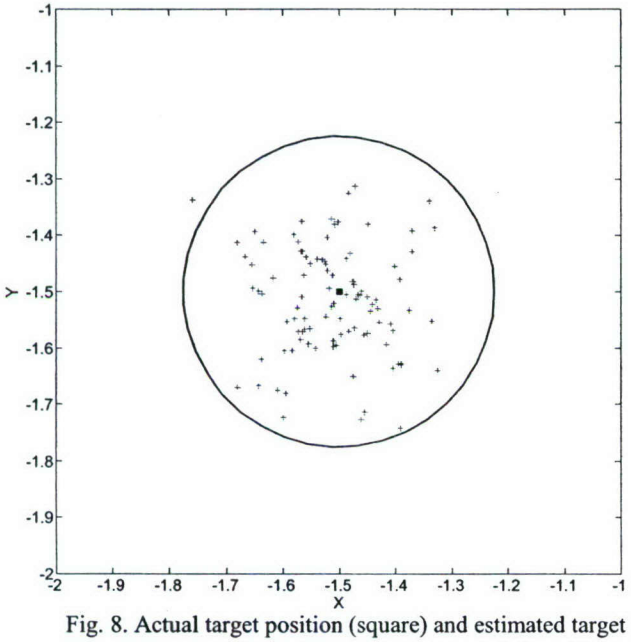


Fig. 8. Actual target position (square) and estimated target positions (plus signs) using the two-step approach for 100 Monte Carlo runs. ($x_t = y_t = -1.5$, N=49, P_o=64, n=2, α=2, T=10). The ellipse shows a theoretical 99% confidence region based on the CRLB.

For both the fixed and random target position, the two-step approach gave unbiased estimates of the target position with the mean estimation error within the 99% confidence interval (Tables III and IV).

TABLE III
MEAN ESTIMATION ERROR AND CONFIDENCE INTERVAL FOR THE MEAN FOR THE TWO-STEP APPROACH (FOR FIXED TARGET POSITION OF (-1.5, -1.5) AND 100 MONTE CARLO RUNS)

Mean Error in Estimated x_t	99% C.I. for Mean Error in Estimated x_t		Mean Error in Estimated y_t	99% C.I. for Mean Error in Estimated y_t	
	Lower Bound	Upper Bound		Lower Bound	Upper Bound
-0.0108	-0.0284	0.0067	-0.0164	-0.0354	0.0027

TABLE IV
MEAN ESTIMATION ERROR AND CONFIDENCE INTERVAL FOR THE MEAN FOR THE TWO-STEP APPROACH (FOR RANDOM TARGET POSITION, AND 500 MONTE CARLO RUNS)

Mean Error in Estimated x_t	99% C.I. for Mean Error in Estimated x_t		Mean Error in Estimated y_t	99% C.I. for Mean Error in Estimated y_t	
	Lower Bound	Upper Bound		Lower Bound	Upper Bound
0.0042	-0.0083	0.0167	0.0050	-0.0069	0.0170

The ML surface of Fig. 1 is smooth near the target location. The results of this study confirm that the coarse WA estimate provided in the first stage provides an initial estimate that is near enough to the function's global maximum to facilitate convergence of the deterministic algorithm used in the second stage of the two-stage approach.

6 Conclusion

In this paper, a simple two-stage approach for ML target localization using binary sensor data was presented. This approach may be a step towards facilitating the routine use of the promising ML target localization methods described in [2] and [3] in which the use of quantized sensor data reduces the needed communication bandwidth. In our simulations, a deterministic search with a random starting point frequently failed by converging to a local minimum of the multimodal ML surface of this problem. Simulation results show that the two-stage approach is accurate and would require far fewer computations than alternatives such as stochastic algorithms.

A weight for each timeframe equal to the square of the signal power level required to produce a detection (11) proved to be adequate for the WA step in the two-step approach. However, with this weighting scheme, the performance of the WA estimate varies to some extent with the target location. Other weights might yield improved performance.

The problem of estimating a target location with *a priori* knowledge of the power of the target, P_o (1), was addressed in this paper. The ML localization problem is more complex in situations where P_o is unknown. Extension of the two-stage approach for ML localization in cases where P_o is unknown will be considered in future work.

7 Acknowledgment

Parts of this effort were sponsored by the Department of the Navy, Office of Naval Research. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the Office of Naval Research.

8 References

[1] C. Y. Chong, S.P. Kumar, "Sensor networks: Evolution, opportunities, and challenges," in *Proc. IEEE*, 2003, vol. 91, pp. 1247-1256.

[2] R. Niu and P. K. Varshney, "Target location estimation in wireless sensor networks using binary data," in *Proc. Int. Conf. Acoust., Speech, Signal Process.*, May 2001, pp. 2037-2040.

[3] R. Niu and P. K. Varshney, "Target location estimation in sensor networks with quantized data," in *IEEE Trans. Signal Process.*, December, 2006, vol. 54, pp. 4519-4528.

[4] M.M. Noel, P.P Joshi, and T.C. Jannett, "Improved maximum likelihood estimation of target position in wireless sensor networks using particle swarm optimization," in *Proc. Third Int. Conf. ITNG*, 2006, pp. 274-278.

[5] Y. Bar-Shalom and X. R. Li, *Estimation and Tracking—Principles, Techniques, and Software*. Boston, MA: Artech House, 1993.

Comparing Agent Paradigms for Resource Management in Sensor Networks

Anish Anthony
Department of Electrical and Computer Engineering
The University of Alabama at Birmingham
Birmingham, Alabama, USA

Thomas C. Jannett
Department of Electrical and Computer Engineering
The University of Alabama at Birmingham
Birmingham, Alabama, USA

Abstract *- Management of power and other resources that effect field life is an important consideration in sensor networks. This paper presents and compares alternative agent paradigms applicable for resource management. A modular simulation framework based on object-oriented design was used to generate data for detailed analysis of component interactions and for evaluation of the integrated sensor network system. Monte Carlo simulations were used to compare the number of computations, number of communications, tracking performance, intelligence quotient, and field life for scenarios utilizing different agent paradigms. These comparisons will assist designers of agent-based systems in utilizing agents to their best advantages in scenarios similar to the wide range of those explored in this paper.*

Keywords: *Sensor Networks, Agents, Resource Management, Machine Intelligence Quotient, Simulation.*

1 Introduction

Sensor networks consist of many spatially distributed sensor nodes that can be used to detect and monitor phenomena at different locations. Applications of sensor networks include environmental monitoring, health and hospital related applications, infrastructure security, surveillance, and tracking. This paper considers an application of distributed sensor networks in which a large number of sensor nodes are placed on the ocean floor near the mouth of a port or bay. The network is used for target detection and tracking.

Individual sensor nodes have limited computational capacity and battery power. A node's batteries may be depleted through normal operation, leaving the node useless and possibly weakening the network. With reduced resources, the network may no longer be able to deliver the required level of performance. Thus, field life is an important consideration in designing sensor networks [1].

Agent-based systems are a new and robust paradigm for designing flexible architectures for systems with disparate needs. An agent is an entity that acts in the place of another, with its authority, in order to bring about a desired result. Agents are capable of carrying out goals alone or they can work together as a part of a larger community. Agents provide a level of abstraction for achieving goals in a system, thereby potentially simplifying the design of a complex system. Agents are gaining popularity because of their flexibility, modularity, and general applicability, especially in the field of distributed computing [2].

Agents have much to offer in sensor networks. Agents can be added, removed, and modified to offer great flexibility in the design of distributed systems. Higher fault-tolerance and better load balancing can be achieved by using multiple agents. Applications of agents in sensor networks include target tracking, wild fire monitoring, traffic management systems etc. [3-5]. However, since this is an emerging field, the literature offering guiding principles or strategies for utilizing agents in sensor networks is sparse.

The objective of this work is to present and compare different agent-based paradigms developed to achieve the goal of managing resources and increasing field life in the undersea distributed sensor network. Simulations were used to evaluate the performance of the different paradigms. This comparison provides an increased understanding of how the performance of the network changes with the number, functionality, and presence of agents at different levels in the network hierarchy. Field life, number of computations, number of communications, tracking performance, and a measure of intelligence (IQ) are network specific performance measures used to compare the different paradigms.

Agents have been used for resource management and improving field life in sensor networks. However, we are unaware of other work that compares paradigms based on the number of agents, agent functionality, presence of agents at different levels in the network hierarchy, and the intelligence quotient of the system. Even though a particular undersea application is considered in this paper, many of the results from this research can be applied to any agent-based sensor network system.

This paper is organized as follows. Section II describes the agent architecture and alternative agent paradigms that are compared in this paper. The alternative agent-based sensor network simulation scenarios are presented in section III. The simulation framework is presented in section IV and the performance metrics used to compare the different paradigms are described in section V. The results are discussed in section VI followed by the conclusion in section VII.

2 Agent-based sensor network system

Fig. 1 shows the field layout of an exemplary sensor network that detects and tracks a target through the field. A grid layout is assumed although other field layouts could also be easily implemented. The network has four master nodes (large circles), nine cluster nodes (black dots) and seventeen sensor nodes (small circles) per cluster. Only one master node is active in the network at any time. The sensor nodes report measured range and bearing of the target to cluster nodes that perform local data fusion. The active master node (diamond) then gathers the data from the cluster nodes, performs global data fusion, generates an estimate of the target position, and tracks the target through the field. The active master node may also provide information about the target and the network to an external command center. When one master node fails, it may be replaced by one of the redundant master nodes that is activated to carry out the master node functions.

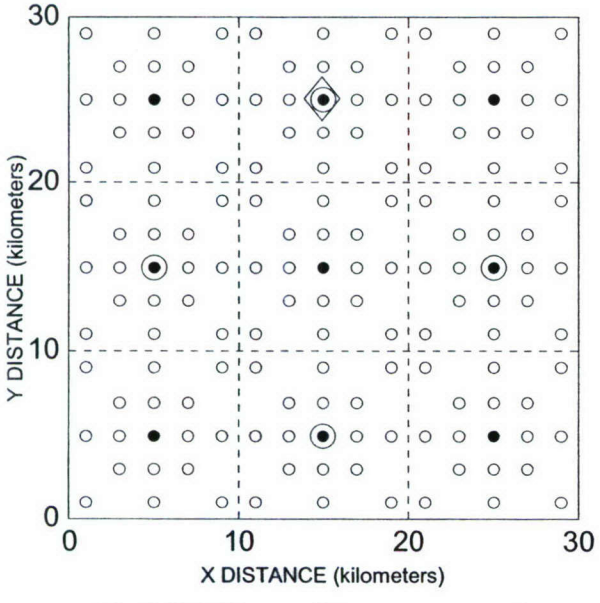


Fig. 1. Field layout of the sensor network.

In a simple agent architecture, the agent forms a wrapper around the node and perceives the environment from data collected by the sensors, messages from other agents, or events that occur within the node. Depending on the goal that has been set for the agent, the agent applies some condition - action rules. These rules help the agent to make a decision on what action has to be taken in order to achieve the goal. Fig. 2 shows the hierarchy of the sensor network architecture in which master agents (MA) are present on the master node and cluster agents (CA) are present on the cluster nodes. For the undersea distributed sensor network application under consideration, agents will be used for target tracking, control, coordination, and reconfiguration of the system and its resources in order to maximize the field life.

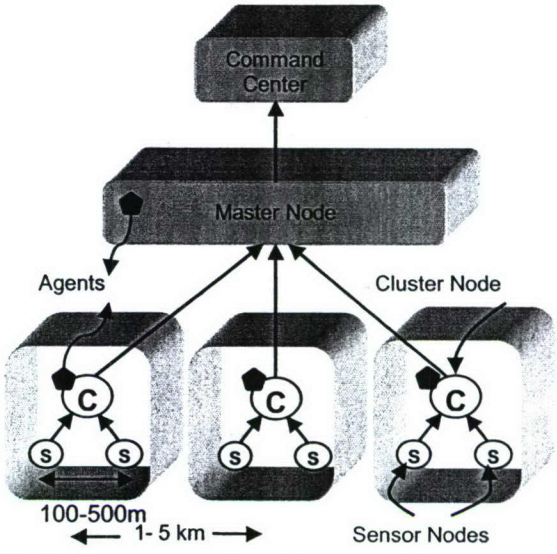


Fig. 2. Sensor network architecture with agents on the cluster nodes and the master node.

2.1 Paradigms based on numbers of agents

Depending on the number of stationary agents in the system, two strategies can be implemented in a distributed sensor network system. These two strategies having agents are reviewed below.

2.1.1 Single stationary agent system: In this scheme, a single agent located on the master node is responsible for data fusion, target tracking, and control and coordination. Centralized single agent systems are easy to design but possess the disadvantages of centralized systems, including poor survivability.

2.1.2 Multiple stationary agent system: In multiple agent systems, individual goals may be the same or different for the agents, but the agents work together to achieve the global goals for the system. In a multi-agent system, there may or may not be direct interaction between agents. For a multi-agent system in which the agents interact with each other, the system design becomes complex. In a multi-agent system, agents may be located on the master node and on the cluster nodes. In this work, the master agents are responsible for global data fusion and tracking and for control and coordination of the clusters. The cluster agents are responsible for local data fusion and for control and coordination of the local sensors.

2.2 Paradigms based on agent functionalities

Different algorithms may be employed by the agents for data fusion, tracking, control, and coordination. The agents have the ability to intelligently decide which algorithms to use and when to use them in order to achieve the goal set for the agent. Some of these algorithms developed in previous work are briefly reviewed below.

2.2.1 Agents for data fusion and tracking: A Kalman filter tracking algorithm for a distributed sensor network system [6] has been adapted to serve in this application and was encapsulated in the agent-based systems to achieve the goals of data fusion and target tracking. Agents provide intelligence by deciding when to track, which tracking algorithms to use, and how frequently to run these tracking algorithms. If there are communication delays in the network, agents can also decide whether to skip the delayed and old data or use the delayed data to generate a new target track. If there is a slow moving target, then the agents can run the tracking algorithms infrequently and save node power. Some tracking algorithms may have better resolution but may be more computationally intensive. Thus, the agents can use tracking algorithms efficiently to increase the field life of the system while maintaining the required tracking performance.

2.2.2 Agents for control and coordination: Intelligent agents can be used to implement control and coordination strategies in order to maximize the useful life of the field. Agents for control and coordination can gather information about the battery life of the nodes in the network and modify communication routes through the network. Changing communication routes can help to decrease communication delays in the system as well as to manage the power consumption of the nodes that are being used for communication [7]. If a node dies in the system, agents can switch functions from the dead node to nodes that still have adequate battery energy remaining [1]. Agents can also be used to selectively turn on and off nodes in the system. Thus, sensor nodes that are not in the vicinity of the detected target can be temporarily switched off to save battery energy.

3 Scenario simulations

Agent-based paradigms developed using multiple agents, different agent functionalities, and agents at different levels in the system hierarchy are described below. Only the most significant scenarios considered are presented in this paper.

3.1 Scenario 1

No agents are employed in scenario 1. Master, cluster, and sensor nodes operate continuously.

3.2 Scenario 2

One master node and one master agent are used in scenario 2. The master agent runs the computationally intensive data fusion and tracking algorithms periodically in a mode referred to as Timed Tracking (TT). In TT, the measurements from the sensors are used periodically to estimate the target position. In between successive measurements, the tracking algorithm does a time update on the estimate to predict the target position. The battery energy of the master node is conserved using TT. The tracking periodicity is determined within the agent using a fuzzy logic algorithm. The fuzzy rules are developed such that the periodicity depends on the velocity of the target and the battery life remaining on the master node. For a master node having full or almost full battery reserve, the tracking is done less frequently (every 30-35 minutes) when a target is moving slowly through the field (0-10 knots/hr) than when the target is moving fast (Fig. 3). Also, tracking is performed less frequently as the battery level of the master node decreases.

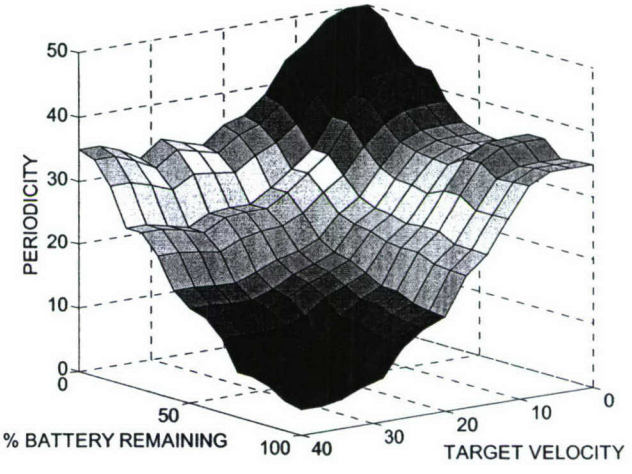


Fig. 3. Surface plot for the timed tracking (TT) fuzzy logic algorithm.

3.3 Scenario 3

In scenario 3, only one master node is present in the system and no master agents are present. Cluster agents are present on each cluster. The cluster agents perform data fusion and track using TT. The periodicity of fusion is determined as for scenario 2, but in this case TT is run only at the cluster level in the system hierarchy.

3.4 Scenario 4

In this scenario, four master nodes and four master agents are present in the system but only one master node and agent is active at a time. Master agents use TT for periodic tracking. When the battery of the current master node is about to die, the master agent at that node transfers the control to a master node in the nearest proximity to the target using a scheme referred to as Control and Coordination scheme 1 (CC1).

3.5 Scenario 5

Four master nodes and four master agents are used in scenario 5 but only one master node and agent is active at a time. The master agents use CC1 and a covariance tracking mode (CT). In CT, the master agent running the Kalman filter tracking and fusion algorithm compares the target position estimate covariance to a threshold. If the covariance is below the threshold, the master agent stops the data fusion process until the covariance increases above the threshold. The master agent forces the tracking algorithm to

perform a time update on its estimates until the covariance of the estimate goes above the threshold. The threshold covariance level is adjusted to attain a desired tracking performance. For a tracking performance having a low root mean square (RMS) error, the threshold covariance level will be smaller than that set to obtain a larger RMS error. In this scenario the master agent also has the ability to instruct the cluster agents to turn the cluster nodes on or off. Only clusters adjacent to the cluster where the target is estimated to be present are turned on by the master agent using a scheme referred to as Control and Coordination scheme 2 (CC2). In addition, cluster agents on each cluster node perform control and coordination. When the master agent instructs the clusters to turn on or off, the cluster agents use Control and Coordination scheme 3 (CC3) to instruct the sensors to turn on or off in order to minimize the battery power consumption at the different nodes in the system.

3.6 Scenario 6

Four master nodes and four master agents are used in scenario 6 but only one master node and agent is active at a time. Cluster agents are present on all cluster nodes. The master agents use the proximity to the target-switching scheme (CC1) to switch between master nodes. The master agent turns off clusters that are far away from the target (CC2). Cluster agents use CT for tracking and CC3 to turn the sensors within each cluster on and off.

Table 1 summarizes the number of master and cluster nodes, number of master agents (*nma*), number of cluster agents (*nca*), and algorithms employed by the agents in different scenarios.

TABLE 1
Scenario descriptions

	SCENARIO					
	1	2	3	4	5	6
Master nodes	1	1	1	4	4	4
Cluster nodes	9	9	9	9	9	9
nma	0	1	0	4	4	4
nca	0	0	9	0	9	9
MA mode						
TT		X		X		
CT					X	
CC1				X	X	X
CC2					X	X
CA mode						
TT			X			
CT						X
CC3					X	X

4 Simulation setup

A framework for simulating agent-based scenarios was developed to utilize the advantages of object-oriented programming techniques. This framework allows the sensor network to be scaled easily. Nodes can be added or removed, different communication and battery models can be simulated, and different algorithms can be easily employed within the agents to facilitate simulation of a large number of scenarios in a short period of time. This framework is not restricted by this application but can be easily adapted for simulating agent-based sensor networks for other applications.

Fig. 4 shows the object diagram for the simulation framework. Agents are present on the master and the cluster nodes. The communication model represents the communication between nodes. A simple outbox-inbox communication scheme transfers the data between the nodes. The data is transferred from the outbox of the transmitting node to the inbox of the receiving node. A localized optimization scheme is used to route the data through the network [7]. Localized optimization uses the fact that cluster nodes can acquire the residual energy levels of sensor nodes using minimal communication within the cluster. This information is used by the cluster for localized network reconfiguration. The communication model is also responsible for introducing communication delays in the system due to computational and communicational loads.

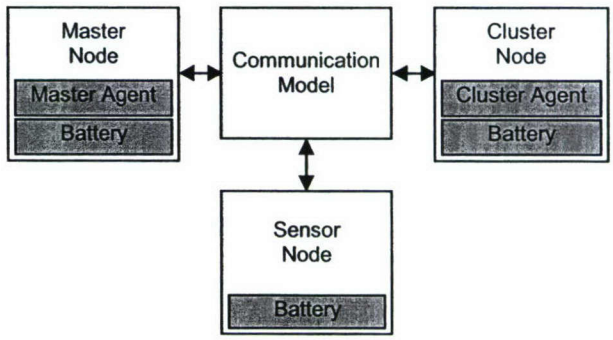


Fig. 4. Object diagram of the simulation framework.

5 Performance metrics

The performance of each agent-based paradigm was evaluated using 30 trials of Monte Carlo simulations. For each trial the target was set up with random initial position and velocity. The following network specific performance measures were used to compare the different paradigms.

5.1 Life

The field life of the network was defined to be the simulation duration before the battery charge of any sensor, cluster, or master node is depleted. In scenarios having multiple master nodes, the battery charge must be depleted at all master nodes in order for the simulation to end due to loss of the master node function.

5.2 Number of computations

Data fusion, tracking, control, and coordination algorithms are computationally intensive. A counter accumulates the number of computations performed at each node at every time step. Proportionate weights are assigned to the computational elements and the node batteries are drained accordingly.

5.3 Number of communications

Communications utilize battery resources. Information flows from the sensors to the cluster nodes and then to the master node. The control signals flow from the master node to the cluster nodes and then to the sensors. A counter accumulates the number of communications taking place in the network. Proportionate weights are assigned to the communication tasks and the node batteries are drained accordingly.

5.4 Tracking performance

For each Monte Carlo run, the error between the actual target position and the estimated target position is calculated. The mean of the root mean square (RMS) errors over 30 Monte Carlo simulation runs is used to reflect the tracking performance of the system. This tracking performance measure allows tradeoffs between field life for the network and tracking performance to be examined.

5.5 Machine intelligence quotient

Many different agent scenarios can be envisioned for sensor networks and other applications. However, there are no guidelines for assessing the performance benefits that would be derived from more complex realizations. Considering the different agent-based nodes as intelligent machines, a measure of the machine intelligence quotient (MIQ) was developed to facilitate comparing agent alternatives of different complexities [8]. The MIQ gives a theoretical measure for quantifying the benefits of the decisions made by agents to achieve the goals of the sensor network. The measure of the MIQ of an agent-based distributed sensor network was developed by removing the contributions of the human element from the method for determining the MIQ of a human-machine cooperative system [9]. We have further refined the measure of MIQ in this paper.

In the sensor network under consideration, agents provide intelligence with a goal of increasing the network field life without adversely affecting performance. Even without agents, the system performs certain tasks that are achieved with what will be referred to as base system intelligence (BSI). The distributed sensor network system is also decomposed into a number of intelligent machines (IMs) working together to achieve system goals, with each IM encompassing an agent. Each IM in the system is assumed to perform 8 tasks (detect, observe, identify, interpret, evaluate, define, select actions, execute actions) in making a decision. The MIQ of the total system is calculated as the sum of the BSI and MIQs of each intelligent machine in the system

$$MIQ_{total} = BSI + \sum_{ma=1}^{nma} MIQ(ma) + \sum_{ca=1}^{nca} MIQ(ca) \quad (1)$$

where
BSI is the MIQ contributed by the base system
MIQ(ma) is the MIQ contributed by master agent *ma*
MIQ(ca) is the MIQ contributed by cluster agent *ca*
nma is the number of master agents, and
nca is the number of cluster agents.

The MIQ contributed by an agent is calculated using

$$MIQ = \sum_{i=1}^{n} r_i \quad (2)$$

where n is the number of tasks performed by each agent (n=8) and r is the intelligence cost required to perform a task. The task intelligence costs, r_i, for each task performed are presented in Table 2. The task intelligence costs were assigned by analyzing the algorithms employed by the agents and the decisions that were made.

TABLE 2
Task intelligence costs

Task	Task No i	No Agent *BSI*	MA TT r_i	MA CT r_i	MA CC1 r_i	MA CC2 r_i	CA TT r_i	CA CT r_i	CA CC3 r_i
Detect	1	5	3	5	2	5	10	10	12
Observe	2	3	2	2	2	7	2	2	7
Identify	3	7	2	2	2	8	2	2	8
Interpret	4	10	5	8	3	8	12	15	8
Evaluate	5	0	10	12	6	6	10	12	6
Define	6	0	5	5	5	5	5	5	5
Select	7	0	5	5	5	5	5	5	5
Execute	8	0	10	10	15	18	10	10	18
Sum =		25	42	49	40	62	56	61	69

For the scenarios explored in this paper, the intelligence required by the IMs to perform two or more decision-making functions is independent and so the total machine intelligence quotient is given as

$$MIQ_{total} = BSI + \sum_{ma=1}^{nma} \sum_{f=1}^{nf} MIQ(ma,f) + \sum_{ca=1}^{nca} \sum_{f=1}^{nf} MIQ(ca,f) \quad (3)$$

where *nf* is the number of decision-making functions performed by the IM. From (2) and (3)

$$MIQ_{total} = BSI + \left\{ \sum_{i=1}^{n} r_i; \forall f; \forall ma \right\} + \left\{ \sum_{i=1}^{n} r_i; \forall f; \forall ca \right\}. \quad (4)$$

6 Results and discussion

Fig. 5 shows the average number of computations and communications performed at every simulation instant for each scenario. Fig. 6 shows the percentage of trials in which the simulation ended due to the field life being limited by the loss of a sensor, cluster, or master node. Fig. 7 shows the average field life obtained for each scenario. Fig. 8 shows the tracking performance achieved for the different scenarios.

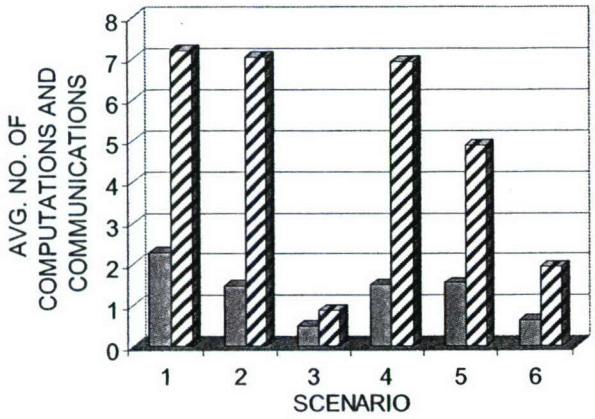


Fig. 5. Average number of computations (grey) and communications (striped) per simulation instant for each agent scenario in Monte Carlo simulations (N=30).

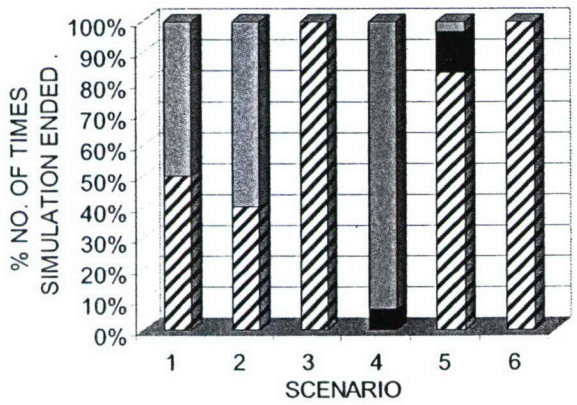


Fig. 6. The percentage of trials for which the simulation ended due to field life being limited by loss of a sensor node (grey), a cluster node (black bar), and all master nodes (striped) for each agent scenario in Monte Carlo simulations (N=30).

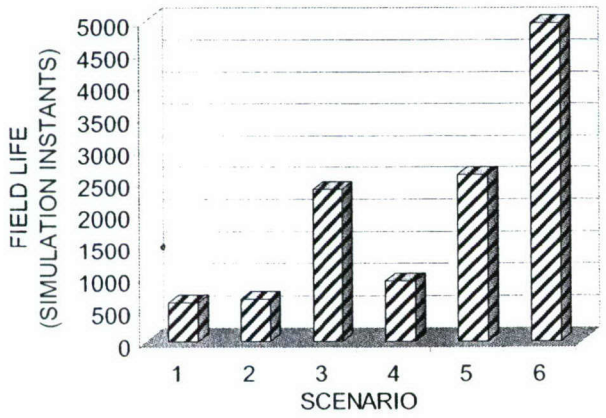


Fig. 7. Average field life (simulation instants) achieved for each agent scenario in Monte Carlo simulations (N=30).

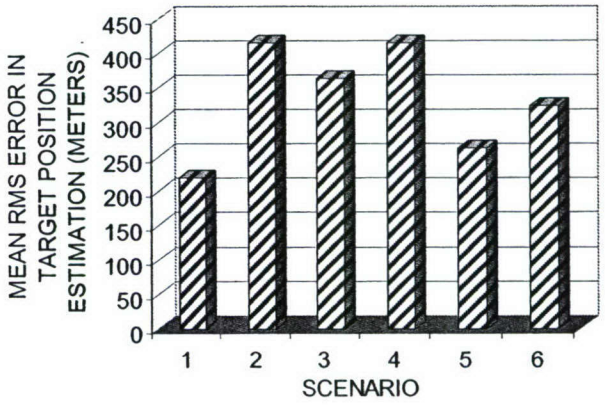


Fig. 8. Mean RMS target position estimation error achieved for each agent scenario in Monte Carlo simulations (N=30).

Scenarios 1, 2, and 3 used 1 master node and 9 cluster nodes. In scenario 1, there were no agents present in the system. The simulations ended due to loss of a sensor node 50% of the time and due to loss of a master node 50% of the time (Fig. 6). No resource management strategies were utilized. The tracking performance was good (Fig. 8) but since there is a large communications overhead (Fig. 5), the field life was low (Fig. 7).

In scenario 2, the master agent allowed the master node to execute the high-level fusion and tracking algorithms periodically (TT) in order to reduce the number of computations performed at the master node (Fig. 5). However, since the sensor and cluster nodes were on continuously as in scenario 1, the number of communications in the system was high (Fig. 5) and the field life was low (Fig. 7) even though the tracking performance suffered with TT (Fig. 8). In scenario 2, the simulations ended due to loss of a sensor node 60% of the time and due to loss of the single master node 40% of the time (Fig. 6).

In scenario 3, cluster agents were used to control the periodicity of execution of the intermediate-level fusion algorithms running on cluster nodes (TT). Each cluster node gathered data from its sensors periodically to reduce the number of sensor-to-cluster communications. The master node periodically received target position estimates from cluster nodes, reducing the number of cluster-to-master node communications. The communications reduction increased the field life (Fig. 7) without much of a reduction in tracking performance (Fig. 8). Thus, just by making the agent perform TT at the cluster level instead of at the master level, a considerable improvement in field life was achieved (Fig. 7).

Scenarios 4, 5, and 6 used 4 master nodes and 9 cluster nodes. As in scenario 2, in scenario 4 the master agent employed TT. However, since the sensor and cluster nodes were on continuously, the number of communications in the system was high (Fig. 5) and the tracking performance also suffered with TT (Fig. 8). Scenario 4 used the CC1 scheme effectively to switch the master node function from one master node to another, such that the field life ended due to due to the loss of a sensor or cluster node and not a master

node (Fig. 6). By using redundant master nodes, scenario 4 was able to achieve a higher field life compared to scenario 2. But because of the higher number of communications (Fig. 5), scenario 4 had a field life lower than scenario 3 (Fig. 7).

Scenario 5 used the CT tracking mode at the master level to save power while preserving tracking performance (Fig. 8). In scenario 5, the master agents could decide to put a cluster node into hibernation (CC2), and cluster agents, in turn, could decide to put the sensor nodes into hibernation (CC3). Thus, the power saved with CT and that saved using coordination modes CC1, CC2, and CC3 increased field life (Fig. 7).

Scenario 6 used CT at the cluster level and used coordination modes CC1, CC2, and CC3 to further increase field life (Fig. 7) at the expense of a slightly degraded tracking performance (Fig. 8).

With TT, the tracking performance degrades gracefully because the rate at which the fusion and tracking algorithms are run decreases with the decrease in power reserves. However, TT is not generally applicable for use because of its reduced accuracy. In comparison, in CT mode, the system always maintains the tracking performance that is set as the threshold. Thus, the mean of the RMS errors for CT in scenarios 5 and 6 was less than that for TT in scenarios 2, 3 and 4 (Fig. 8).

The use of coordination modes CC1, CC2, and CC3 in scenarios 5 and 6 resulted in a more balanced utilization of power resources in the network than other scenarios. A balanced utilization of power resources is important since it keeps a node from dying prematurely to end the field life.

The MIQs evaluated using (4) for the different agent-based scenarios are presented along with field life in Table 3. The number of agents in the system is an important factor in determining the MIQ for the total agent-based sensor network system. Scenarios 5 and 6 had the highest number of agents and hence the highest MIQ. The field life performance is higher for systems with high MIQs (Table 3 and Fig. 7). The MIQ represents a capability to perform, but the simulation parameters and operational challenges may not allow the capabilities of a system having a high MIQ to be fully utilized.

TABLE 3
MIQ and average field life for different scenarios.

	SCENARIO					
	1	2	3	4	5	6
MIQ	25	67	529	353	1250	1603
Field Life	603	667	2368	840	2592	4063

7 Conclusion

This paper presents a comparison of several agent-based scenarios in a distributed sensor network. Agents with different abilities were used at different levels in the system hierarchy. Agents were used to maximize the system data fusion and target tracking performance while reducing the power consumption to increase the field life of the sensor network. The simulation framework demonstrated the scalability and flexibility of an agent-based system where task-specific agents are added or removed to simulate different scenarios with ease.

The results demonstrate the need to make the appropriate algorithms available at the levels in the hierarchy where they can be of the most benefit. Scenarios having high MIQs generally achieved a high performance, especially in cases where the agents were given the capability to use the best tracking, control and coordination algorithms.

8 Acknowledgment

Parts of this effort were sponsored by the Department of the Navy, Office of Naval Research. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the Office of Naval Research.

9 References

[1] S. V. Chandrachood, A. Anthony, and T. C. Jannett, "Using resource based modeling to evaluate coordination schemes in wireless sensor networks," in *Proc. IEEE SouthEastCon 2006*, Memphis, TN, pp. 91-97.

[2] C. C. Hayes, "Agents in a nutshell - a very brief introduction," *IEEE Trans. on Knowl. Data Eng.*, vol. 11, pp. 127-133, Jan/Feb 1999.

[3] S. Hussain, E. Shakshuki, A. W. Matin, and A. R. Matin, "Collaborative agents for data dissemination in wireless sensor networks," in *20th Int. Symp. High Performance Computing Systems*, p. 16, 2006.

[4] C. Fok, G. Roman, and C. Lu, "Mobile agent middleware for sensor networks: An application case study," in *Proc. 4th Int. Conf. Inform. Process. in Sensor Networks*, Los Angeles, CA, 2005, pp. 382-387.

[5] F. Y. Wang, "Agent-based control for networked traffic management systems," *IEEE Intell. Syst.*, vol. 20, pp. 92-96, Sept./Oct. 2005.

[6] R. Kasthurirangan, "Comparison of architectures for multi-sensor data fusion in deployable autonomous distributed systems," M.S. thesis, Dept. Elect. and Comp. Eng., The Univ. of Alabama, Birmingham, 2003.

[7] R. Praveenkumar and T.C. Jannett, "Investigation of routing protocols in a sensor network using resource based models," in *Proc. IEEE SoutheastCon 2007*, Richmond, VA, pp. 29-34.

[8] A. Anthony and T. C. Jannett, "Measuring machine intelligence of an agent-based distributed sensor network system," in *Proc. Int. Joint Conf. CISSE*, Dec 2006.

[9] H. J. Park, B. K. Kim, and K. Y. Lim, "Measuring the machine intelligence quotient (MIQ) of human-machine cooperative systems," *IEEE Trans. Syst. Man Cybern. A., Syst. Humans*, vol. 31, pp. 89-96, Mar. 2001.

Localized Performance-Guided Reconfiguration for Distributed Sensor Networks

Parag P. Joshi
Dept. of Electrical and Computer Engineering
The University of Alabama at Birmingham
Birmingham, AL 35294-4461
paragj@uab.edu

Thomas C. Jannett
Dept. of Electrical and Computer Engineering
The University of Alabama at Birmingham
Birmingham, AL 35294-4461
tjannett@uab.edu

Abstract

Sensor networks that can reconfigure to extend their useful operating life while guaranteeing coverage performance are of interest. This paper describes the use of localized Performance-Guided Reconfiguration (PGR) for reconfiguring a distributed sensor network in a target localization scenario using binary sensor data. If a sensor in a cluster becomes unavailable due to a loss of battery power or other reasons, PGR activates redundant sensors as necessary to allow the network to achieve the desired coverage performance. For a simple demonstration of localized PGR, we compare the performance of a multicluster network using localized PGR, a nearest-neighbor reconfiguration approach, and an approach where no reconfiguration of the field is done. Without PGR, coverage performance degraded as sensors were lost. With PGR, every time a sensor failed, the respective cluster was reconfigured to maintain coverage. In this way, clusters made local reconfiguration decisions to achieve the global field coverage performance goal.

1. Introduction

A wireless sensor network (WSN) is a collection of spatially distributed sensing devices that function cooperatively to garner information about an environment or event of interest. Initially motivated by military applications such as battlefield surveillance, WSNs, today, are being applied in areas such as home automation, traffic control, process monitoring, precision agriculture, emergency response, and structural health monitoring [1].

Power is the scarcest resource available to sensor nodes. Since WSNs are installed in hostile or inaccessible environments, algorithms for control, coordination, routing, and communication need to be energy-efficient. Over time, node batteries can weaken and eventually die, leaving the nodes powerless, and the network crippled. In order for the network to be able to regain its operational performance at the desired level, a rearrangement or reconfiguration of resources is imperative.

In a simulation study, repeated rerouting of communications to balance power utilization and to increase the life of the sensor field was accomplished by optimization of a complex cost function using evolutionary computation [2]. Rerouting of communications resulted in a uniform drain of node batteries throughout the network, and thus extended the field life significantly. However, this centralized approach required repeated optimization of a complex cost function and presented a significant communications overhead in acquiring the data needed for optimization.

In another study, a mobile node approach was employed, wherein a few mobile nodes were deployed along with static nodes [3]. In the event that a static node failed, a coverage hole formed. A fuzzy distributed decision making algorithm was then used to determine a mobile node location that would maximize a utility function depending on connectivity and coverage gains.

Performance-guided reconfiguration (PGR) is an algorithm that can be employed to reconfigure a sensor network to meet the desired performance goals when sensor failures occur. When a sensor fails, PGR identifies candidates for replacing the failed sensor from a set of available redundant sensors, and uses a performance-based cost function to select the candidates to be activated [4]. In order to make judicious use of resources in a distributed system, localized algorithms that do not require global information are needed [5]. Some initial simulation results of localized PGR, in which PGR is applied at the cluster level in a multicluster sensor network, were described in [6].

In this paper, the work in [6] was extended by using simulations to study localized PGR and to compare sensor network performance using localized PGR, no reconfiguration (NR), and a simple nearest-neighbor reconfiguration (NNR) strategy. In the case of NR, as the name suggests, the performance of the network was observed to deteriorate as sensors continued to become unavailable since no redundant sensors were available to be used as replacements. In case of NNR, the nearest redundant sensor was always chosen to be activated, but since there was no means to compare how the performance would be affected by the choice

of redundant sensor to be awakened, the network performance remained below par.

This paper is organized as follows. First, the maximum likelihood (ML) method of target localization using binary sensor data is reviewed. Next, an exemplary sensor network is described, and the localized PGR method is presented. The simulation methods are described. Finally, results comparing localized PGR with NR and NNR strategies are presented.

2. Target Localization

Target localization for position estimation or tracking is an important application of WSNs. Target localization methods include an approach using binary sensor data for which a ML estimator and its Cramer-Rao lower bound have been derived [7]. In this approach, each sensor makes a binary decision about a target's presence by comparing the measured signal strength to a threshold, and communicates a one-bit message to a fusion center. The fusion center uses the binary information received from all the sensors, along with *a priori* information about the positions of the sensors, to localize the target through nonlinear optimization of a highly complex multimodal function. Recently, this approach has been extended for localization based on quantized data [8]. These methods are attractive because they facilitate accurate target localization based on the transmission of binary or multibit quantized data, which requires limited communication bandwidth. The maximum likelihood (ML) estimation framework for target localization using binary data presented in [7] and reviewed in the following is utilized in this paper.

In the model below, signal intensity is assumed to attenuate as the distance from the target to a sensor increases

$$a_i = \sqrt{\frac{P_o}{1+\alpha\, d_i^{\,n}}} \tag{1}$$

where d_i is the distance from the target to the ith sensor, a_i is the signal amplitude at the ith sensor, $\sqrt{P_o}$ is the signal amplitude at the ith sensor when d_i is zero, α is a constant, n is the signal energy decay exponent. Sensor parameters used in this paper are $n = 2$, $\alpha = 2$, and $P_o = 64$. The distance from the target to the ith sensor is given by

$$d_i = \sqrt{(x_i - x_t)^2 + (y_i - y_t)^2} \tag{2}$$

where (x_i, y_i) are the coordinates of the ith sensor and the target location is given by (x_t, y_t). The signal a_i is corrupted by standard Gaussian noise ω_{ij} that is independent across sensor i and time frame j

$$s_{ij} = a_i + \omega_{ij} \tag{3}$$

Each sensor makes a binary decision depending on the signal s_{ij} satisfying a local sensor threshold η_{ij} during timeframe j and transmits its decision to a higher-level fusion node where it is used in target localization. After collecting the decisions I_{ij} from all N sensors for all T timeframes, the higher-level node estimates the parameter vector including the target position $\theta = [x_t\, y_t]$ by maximizing a log likelihood function with respect to θ. An estimator based on the Nelder-Mead search method was used to maximize the log likelihood function, with the search starting at an initial estimate of θ computed as a weighted average of the positions of the reporting sensors in a timeframe.

The Cramer Rao Lower Bound (CRLB) for an unbiased ML estimate is given by

$$E\left\{\left[\hat{\theta}(I)-\theta\right]\left[\hat{\theta}(I)-\theta\right]^T\right\} \geq J^{-1} \tag{4}$$

where J is the Fisher Information Matrix. Details of the Fisher Information Matrix are given in [7]. The covariances of the errors in estimates of (x_t, y_t) are bounded by the (1,1), (2,2), and (1,2) elements in the J^{-1} matrix, respectively

$$\begin{aligned} var(\hat{\theta}_1) = var(\hat{x}_t) &\geq {J_{11}}^{-1} \\ var(\hat{\theta}_2) = var(\hat{y}_t) &\geq {J_{22}}^{-1} \\ cov(\hat{\theta}_1, \hat{\theta}_2) = cov(\hat{x}_t, \hat{y}_t) &\geq {J_{12}}^{-1}. \end{aligned} \tag{5}$$

A lower bound on the variance of the overall target position estimate $\hat{D}$ can then be computed as

$$var(\hat{D}) = var(\hat{\theta}_1) + var(\hat{\theta}_2) + 2\,cov(\hat{\theta}_1, \hat{\theta}_2) \tag{6}$$

where $D = \sqrt{{x_t}^2 + {y_t}^2}$ is the Frobenius norm. Thus, the lower bound for the root-mean square (RMS) error in the overall target position estimate is $\sqrt{var(\hat{D})}$. Computing the lower bound on RMS error across many target locations within a sensor field aids in assessing the performance of alternative network configurations.

3. The Sensor Field Layout

Consider a cluster with 49 active sensors (Fig. 1) placed at intersecting points of a uniform grid with unit spacing.

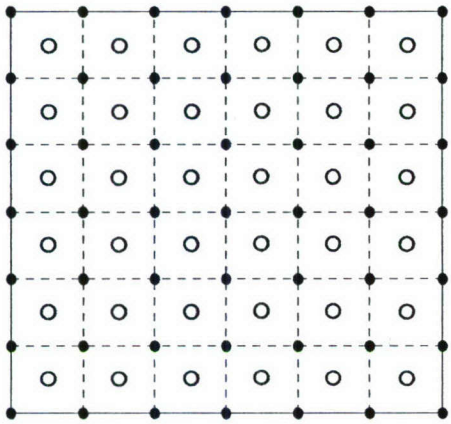


Figure 1. Sensor layout in a cluster. Solid circles show active sensors placed along a uniform grid of unit spacing, and white circles represent redundant sensors uniformly placed between active sensors.

Within each cluster, redundant sensors were uniformly placed at positions between the active sensors. Nine such clusters were grouped together to form a field of 441 active sensors, and 324 redundant sensors. The clusters were positioned to overlap with neighboring clusters such that the nominal sensor field would satisfy a performance criterion for all target locations within the coverage area, as shown bounded by the dotted line in Fig. 2.

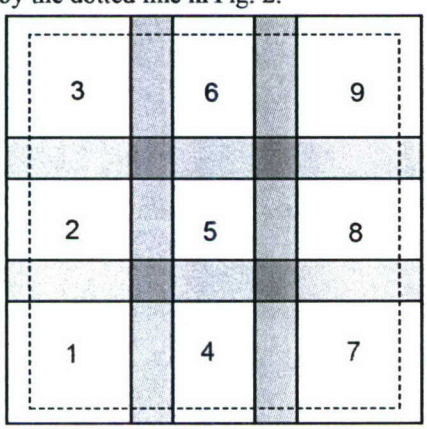


Figure 2. Layout of the field with overlapping clusters. The area of overlap, shown in gray, is 1.25 units wide. The area inside the dotted line is the coverage area.

The coverage area of the field was a square of side 10.5 distance units. The performance criterion was set to 0.155 distance units, which was the maximum allowable value for the computed CRLB on the RMS error in the overall target position estimate, $\hat{D}$ for every point within the coverage area of the field. The overlap between a cluster and its neighbors was such that the performance criterion would be met by the field if the performance criterion of 0.155 distance units was met within the coverage area of each cluster (Fig. 3). In this way, the targeted global field performance would be met through reconfiguration done to meet local performance.

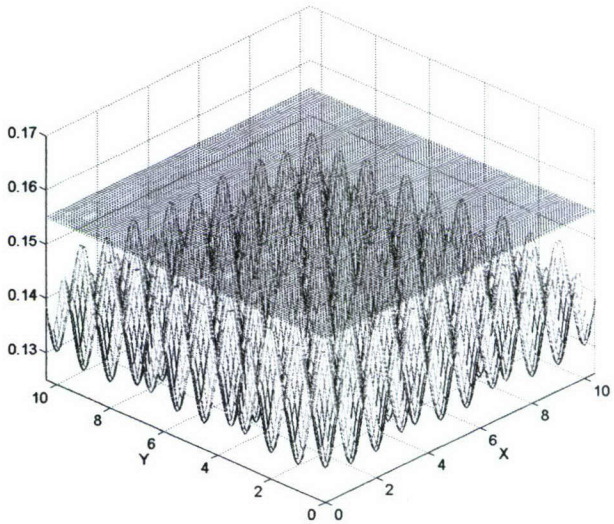


Figure 3. Computed CRLB for the RMS error in the estimate of *D* versus *x* and *y* for the nominal field coverage area when no sensor has failed. The performance criterion of 0.155 distance units for the RMS error is shown as a plane.

4. Localized PGR

PGR works by activating redundant sensors to replace a failed sensor. In this study, the PGR algorithm was applied locally within the cluster that had a sensor fail. Initially, multiple sets of redundant inactive sensors that could be activated to meet the performance criterion within the cluster are identified as candidates for replacing the failed sensor. A cost function that weighs additional criteria of interest is then used to select the candidate sensor set that optimized cluster performance as described in [4].

4.1 Cost Analysis

Since there can be multiple sets of candidate sensors available to replace a lost sensor, making decisions about which candidate set to activate based on additional criteria of interest allows the overall network performance to be optimized. A cost function provides the basis for ranking all possible solutions and for adjusting the weights of the individual performance objectives in terms of the total cost for a candidate set S.

$$Cost(S) = N_s c + \sum \gamma(r_i - r) + \eta \sum_{j=1}^{4} \frac{N_j}{R_j} \qquad (7)$$

γ and η are adjustable constants, N_s is the total number of sensors in candidate set S, c is a flat cost per sensor, and r is the radius of a no-penalty search area around the dead sensor. r_i is the distance of candidate sensor i from the dead sensor, N_j is total number of sensors in area j in candidate set S, and R_j is the number of redundant sensors available in area j (area refers to one of four sectors formed by dividing the area around the dead sensor bounded by the search radius r into four parts).

The cost function weighs the **absolute sensor cost** (term 1), the **distance of the candidate sensor from the failed sensor** (term 2), and the **cost of utilized redundant resources** (term 3). Term 2 is applied only if the candidate sensor is outside a no-penalty search area such that r_i-r is positive. A no-penalty search area is an area bounded by the initial search radius. The search radius is increased if sensors in the area bound by the initial search radius do not meet the performance requirement, and term 2 is then nonzero.

5. Simulation Study

Simulations were used to allow the field performance achieved with localized PGR to be compared to that attained with NR and NNR approaches. A test pattern of sensor failures over time was constructed by simulating sensor failures at random times and at random locations within clusters 1, 2, 4 and 5 in the field (Fig. 2). A total of 23 sensors failed over the duration of the simulation, which was 300 simulation instants. The cumulative number of sensor failures at representative simulation instants is listed in Table 1.

Table 1. Number of failed sensors at representative simulation instants

Simulation Instant	Number of Failed Sensors
34	3
77	7
112	10
184	17
229	20
245	22
257	23

The test pattern was applied for each of the NR, NNR, and localized PGR approaches. For NR, a failed sensor was not replaced. For NNR, a failed sensor was replaced by the nearest redundant sensor. With localized PGR, a failed sensor was replaced by the redundant sensor(s) selected by employing the PGR algorithm locally within the cluster having the failed sensor so that the performance criterion was met inside the cluster.

At each simulation instant that a sensor failed, the CRLB was computed over a dense grid of target positions inside the respective cluster after application of each of the NR, NNR, and localized PGR approaches. The percentage of the coverage area not meeting the performance criterion, H, and an approximation to the volume, V, enclosed by the error surface above the performance criterion plane were computed to serve as the performance metrics used to compare reconfiguration approaches.

6. Results

The coverage performance degraded as sensors failed with NR and NNR (Fig. 4, Table 2). NR represented the worst-case performance achieved when sensors failed and there was no replacement. Some improvement was achieved by replacing failed sensors using NNR.

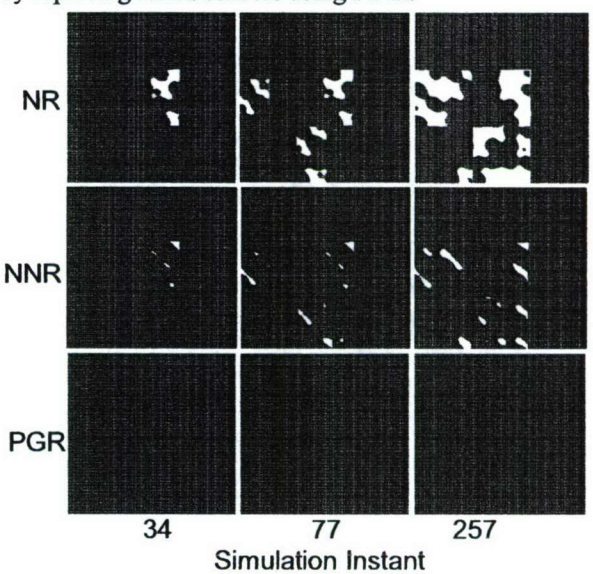


Figure 4. Snapshots of the field performance showing areas not meeting the performance criterion in white for NR, NNR, and localized PGR at simulation instants 34, 77, and 257 from left to right.

In contrast to the performance degradation that resulted with NR and NNR, with localized PGR, activation of redundant sensors to replace failed sensors allowed the performance criterion to be met over the entire duration of the simulations.

Table 2. Comparison of the performance of reconfiguration approaches

	NR		NNR		PGR	
Simul. instant	*H* (%)	*V* ($x10^3 units^3$)	*H* (%)	*V* ($x10^3 units^3$)	*H* (%)	*V* ($x10^3 units^3$)
34	2.4	35	0.5	9	0	0
77	6.9	83	1.9	16	0	0
112	7.9	96	2.1	18	0	0
184	11.3	177	2.7	24	0	0
229	14.1	249	3.0	25	0	0
245	15.0	302	3.8	43	0	0
257	15.8	325	4.0	45	0	0

Without localized PGR (in NNR and NR), the percentage of the coverage area not meeting the performance criterion (*H*) increased as sensors continued to fail, and the degree to which the performance criterion was not met increased such that the size of coverage holes increased and the volume enclosed by the surface above the performance criterion plane (*V*) increased (Fig. 5).

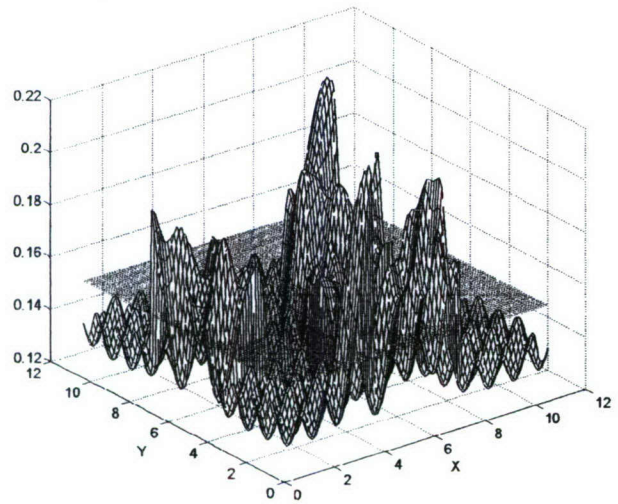


Figure 5. Computed CRLB for the RMS error in the estimate of D versus x and y for the field with no reconfiguration after the failure of 23 sensors over 257 simulation instants. The performance criterion of 0.155 for the RMS error is shown as a plane. A significant part of the coverage area does not meet the performance criterion.

7. Discussion and Conclusions

This paper presented a comparison of the target localization performance of localized network reconfiguration approaches. With localized PGR, the network was reconfigured to replace failed sensors by activating redundant sensors so that the desired performance criterion was always met over the entire coverage area. With NR and NNR, coverage holes formed due to sensor failures and the target localization performance deteriorated.

PGR is an algorithm that can be used to reconfigure a distributed sensor network to allow it to regain its coverage performance after sensors are lost due to battery exhaustion or other reasons. Thus, PGR allows sensor networks to reconfigure in order to extend their useful operating life while guaranteeing coverage performance. In localized PGR, the reconfiguration is done at the cluster level, allowing the sensor network to have the advantages offered by distributed systems. PGR also allows tradeoffs to be made between the performance, resources, and cost benefits from reconfiguration.

In the field layout employed for this study, clusters were made to overlap in such a way that allowed the performance criterion to be met at all target locations inside the coverage area before any sensor failures. The placement of active and redundant sensors was denser in the areas where clusters overlapped than in areas where there was no overlap between clusters. Although this field layout was suitable to demonstrate and compare localized PGR, no attempt was made to optimize the placement of active and redundant sensors. In future work, optimization of sensor placement will allow better use of redundant resources and the overlap between clusters for localized PGR. A more realistic simulation of PGR with resource models mimicking entities and processes that will be present in an actual physical system is another area for future work [9].

8. Acknowledgment

This work was supported in part by the Office of Naval Research under Grant N00014-03-1-0751. Parts of this effort were sponsored by the Department of the Navy, Office of Naval Research. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the Office of Naval Research.

9. References

[1] C. Y. Chong, S.P. Kumar, "Sensor networks: Evolution, opportunities, and challenges", *Proc. IEEE*, August 2003.

[2] M.W. Owen, D.M. Klamer and Barbara Dean. "Evolutionary control of an autonomous field", *Proceedings of the Third International Conference on Information Fusion*, July 10-13, 2000, pp. MoD1-3: 9.

[3] X. Du, M. Zhang, K. Nygard, M. Guizani, and H. Chen, "Distributed decision making algorithm for self-healing sensor networks," to appear in *Proceedings of the 2006 IEEE ICC Conference,* 2006.

[4] P.P. Joshi and T.C. Jannett, "Performance-guided reconfiguration of wireless sensor networks that use binary data for target localization," in *Proceedings of the 2006 ITNG Conference*, 2006, pp. 562-565.

[5] H. Qi, P. T. Kuruganti, and Y. Xu, "The development of localized algorithms in wireless sensor network," *Sensors,* vol. 2, July 2002, pp. 286-293.

[6] P.P. Joshi and T.C. Jannett, "Simulation of localized performance-guided reconfiguration of distributed sensor networks," to appear in *Proceedings of the 2007 ITNG Conference, 2007.*

[7] R. Niu and P. K. Varshney, "Target location estimation in wireless sensor networks using binary data," in *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, May 2001, pp. 2037-2040.

[8] R. Niu and P. K. Varshney, "Target location estimation in sensor networks with quantized data," in *IEEE Transactions on Signal Processing*, vol. 54, December, 2006, pp. 4519-4528.

[9] S. V. Chandrachood, A. Anthony, and T. C. Jannett, "Using resource based modeling to evaluate coordination schemes in wireless sensor networks," in *Proceedings of IEEE SoutheastCon*, Memphis, 2006, pp. 91-97.